Oracle class 1

Types of Data

==============

We have two types of data.

1) Unstructured Data

2) Structured Data

1) Unstructured Data

----------------------------

A data which is not in readable format is called unstructured data.

In general, meaningless data is called unstructured data.

ex:

201 Lakemba SYD NSW AUS

2) Structured Data

------------------

A data which is in readable format is called structured data.

In general, meaning full data is called structured data.

ex:

Unit Locality City State Country

---- -------- ----- ----- --------

201 Lakemba SYD NSW AUS

Management system

=================

Management system is a software which is used to manage the database.

Using management system we can perform following activities very easily.

1) Adding the new data

2) Modifying the existing data

3) Dropping the unnecessary data

4) Selecting the required data

Q) What is the difference between DBMS and RDBMS?

DBMS RDBMS

-------- -----------

It stands for Database Management System. It stands for Relational Database Management

System.

It stores the data in files. It stores the data in tables.

It is not designed to store large amount of It is designed to store large amount of data.

data.

It provides support for single user at a time. It provides support for multiple users at a time.

There is no data security. There is high data security.

It does not support normalization. It supports normalization.

(It reduce code redundency)

Oracle

=======

It is one of the database which is used to store structured data.

It is a RDBMS database.

It is product of Oracle Corporation.

It is classified into two types.

Oracle

|

|--------------------------------------------------------|

SQL PL/SQL

(Structured Query Language) (Procedural / Structured Query Language)

Client-Server Architecture

==========================

Diagram: oracle1.1

Above architecture describes how our frontend data goes to backend.

FrontEnd

---------

The one which is visible to the enduser to perform some operations is called frontend.

ex:

Java,.Net,Python,Perl,D2K and etc.

Communication channel

------------------

It acts like a bridge between frontend and backend.

ex:

JDBC- Java Database Connectivity

ODBC- Open Database Connectivity

PDBC- Python Database Connectivity

BackEnd

--------

The one which is not visible to the enduser but it performs operations based on the instructions

given by frontend is called backend.

ex:

Oracle, MySQL, SQL Server, MongoDB , NoSQL and etc.

SQL

====

SQL stands for Structured Query Language which is pronounce as SEQUEL.

This language is used to interact with oracle database.

It is a command based language.

It is a case insensitive language.

Every command must starts with verb.

Every command must ends with semicolon.

It is developed in the year 1972 by Mr.Codd (by IBM).

Sub languages of SQL

====================

We have five sub languages of SQL.

1) DDL (Data Definition Language)

2) DML (Data Manipulation Language)

3) DRL/DQL (Data Retrieve/Query Language)

4) TCL (Transaction Control Language)

5) DCL (Data Control Language)

1) DDL (Data Definition Language)(parmanent)

-------------------------------

This language is used to maintain the objects in database.

It is a collection of five commands.

ex:

create, alter, drop , truncate and rename.

2) DML (Data Manipulation Language) (temporary)

-----------------------------------

This language is used to manipulate the data which is present in database.

It is a collection of four commands.

ex:

insert, update, delete and merge

3) DRL/DQL (Data Retrieve/Query Language)

------------------------------------

This language is used to retrieve the data from database.

It is a collection of one command.

ex:

select

4) TCL (Transaction Control Language)

---------------------------------------

This language is used to maintain the transaction of database.

It is a collection of three commands.

ex:

commit, rollback and savepoint

5) DCL (Data Control Language)

------------------------------

This language is used to control the access of data to the user.

It is a collection of two commands.

ex:

grant and revoke

Table

=====

Table is an object which is used to represent the data.

Table is a collection of rows and columns.

Data which is present in a table is a case sensitive.

ex:

NO NAME ADD

---------------------------------------

101 | Alan | Florida

---------------------------------------

102 | Jose | Texas

---------------------------------------

103 | Lisa | Chicago

---------------------------------------

Here table contains 3 rows and 3 columns.

Oracle

=======

Version : 10g

Vendor : Oracle Corporation

Website : www.oracle.com/in/products

Software : Expression Edition

Port No : 1521

Username : system (default)

Password : admin

Download link :

<https://drive.google.com/file/d/0B9rC21sL6v0td1NDZXpkUy1oMm8/view?usp=drive_link&resourcekey=0-aKooR3NmAh_eLo_qGw_inA>

class2

Establish the connection with database

======================================

To execute any comment in database or to perform any operation on database we need to

establish the connection with database software.

Once work with database is completed we need to close the connection with database.

ex:

---

SQL> connect

username : system

password : admin

SQL> disconnect

ex:

----

SQL> conn

username : system

password : admin

SQL> disc

ex:

---

SQL> conn system/admin

SQL> disc

create command

====================

It is used to create a table in a database.

syntax:

------

create table <table\_name>(col1 datatype(size),col2 datatype(size),......,

colN datatype(size));

ex:

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

create table dept(deptno number(3),dname varchar2(10),dloc varchar2(10));

create table emp(eid number(3),ename varchar2(10),esal number(10,2),

deptno number(3),job varchar2(10),comm number(8));

Describe command

================

It is used to display the structure of a table.

syntax:

desc <table\_name>;

ex:

desc emp;

desc dept;

desc student;

. Insert command

===================

Insert command is used to insert row/record in a table.

syntax:

------

insert into <table\_name> values (value1,value2,....,valueN);

ex:

insert into student values(101,'raja','hyd');

insert into student values('ravi',102,'delhi'); //invalid

insert into student values(102,'ravi'); // invalid

insert into student values(102,'ravi',null);

Note:

-----

null represent undefined or unavailable.

approach2

---------

insert into student(sno,sname,sadd) values(103,'ramana','vizag');

insert into student(sno,sname) values(104,'ramulu');

approach3

---------

Using '&' symbol we can insert dynamic inputs.

ex:

insert into student values(&sno,'&sname','&sadd');

commit command

==============

It is used to make the changes permanent to database.

syntax:

commit;

dept table

===========

create table dept(deptno number(3),dname varchar2(10),dloc varchar2(10));

insert into dept values(10,'CSE','HYD');

insert into dept values(20,'ECE','PUNE');

insert into dept values(30,'MEC','VIZAG');

insert into dept values(40,'EEE','DELHI');

commit;

emp table

==========

create table emp(eid number(3),ename varchar2(10),esal number(10,2),

deptno number(3),job varchar2(10),comm number(8));

insert into emp values(201,'Alan',9000,10,'Clerk',null);

insert into emp values(202,'Jose',19000,10,'Clerk',500);

insert into emp values(203,'Kelvin',45000,20,'HR',300);

insert into emp values(204,'Nelson',23000,20,'HR',900);

insert into emp values(205,'Lisa',21000,30,'Manager',500);

insert into emp values(206,'Jesicca',37000,30,'Manager',800);

commit;

select command

===================

It is used to retrieve the records from database table.

syntax:-

select \* from <table\_name>;

Here '\*' means all rows and columns.

ex:

select \* from student;

select \* from dept;

select \* from emp;

Projection

----------

Selecting specific columns from database table is called projection.

ex:

select sno,sname,sadd from student;

select sno,sname from student;

select sname from student;

Arithmetic operations

---------------------

In select command we can perform arithmetic operations.

ex:

select sno,sname,sadd from student;

select sno-100,sname,sadd from student;

select sno+100,sname,sadd from student;

Column alias

---------------

A userdefined heading given to a column is called column alias.

Column alias is temperory.

We can create column alias for any column.

ex:

select sno-100,sname,sadd from student;

select sno-100 as SNO,

sname,sadd from student;

select sno as roll\_no,

sname as Name,

sadd as City from student;

Interview Queries

------------------

Q) Write a query to display all employees information from employee table?

select \* from emp;

Q) Write a query to display employee id, employee name and employee salary from emp table?

select eid,ename,esal from emp;

Q) Write a query to display list of tables present in database?

select \* from tab;

Q) Write a query to display logical database name?

select \* from global\_name;

Q) Write a query to display employee id,employee name, employee salary and annual salary from

employee table?

select eid,ename,esal,esal\*12 from emp;

Q) Write a query to display employee id,employee name, employee salary and annual salary as

ANNUAL\_SAL from employee table?

select eid,ename,esal,esal\*12 as ANNUAL\_SAL from emp;

where clause

===============

It is used to select specific records from database table.

syntax:

-------

select \* from <table\_name> where condition;

ex:

select \* from student where sno=101;

select \* from student where sname='raja';

select \* from student where sadd='pune';

Interview Queries

------------------

Q) Write a query to display student information whose is living in hyderabad?

select \* from student where sadd='hyd';

Q) Write a query to display employees information those who are working in 10 department?

select \* from emp where deptno=10;

Q) Write a query to display employee information whose commission is null?

select \* from emp where comm is null;

class3

. update command

=====================

update command is used to update the values in a row.

syntax:

------

update <table\_name> set <col\_name>=value where condition;

ex:

update student set sname='rani' where sno=101;

update student set sname='Alan',sadd='USA' where sno=103;

commit;

rollback;

Note:

-----

If we are not using any where clause then all rows will be udpated.

ex:

update student set sname='raja';

update student set sno=101;

update student set sadd='hyd';

. delete command

================

A delete command is used to delete the rows from database table.

syntax:

------

delete from <table\_name> where condition;

ex:

delete from student where sno=101;

delete from student where sname='ravi';

delete from student where sadd='pune';

commit;

Note:

-----

If we won't use where clause then all rows will be deleted.

ex:

delete from student;

delete from emp;

delete from dept;

Interview Questions

====================

Q) Write a query to terminate all the employees those who are working as a Clerk?

delete from emp where job='Clerk';

Q) Write a query to display employees information whose commision is null?

select \* from emp where comm is null;

Q) Write a query to increment salary by 1000 whose employee id is 201?

update emp set esal=esal+1000 where eid=201;

Logical Operators

=================

Logical operators are used to declare multiple conditions in a query.

We have three logical opreators.

1) AND

2) OR

3) NOT

1) AND

-------

It returns the records only if our condition is true.

All conditions must be from same row only.

ex:

---

select \* from emp where eid=201 AND ename='Alan';

select \* from emp where eid=201 AND ename='Ana'; //no rows selected

select \* from emp where eid=201 AND ename='Jose';//no rows selected

2) OR

---------

It returns the records only if one condition is true.

Here conditions can be from any row.

ex:

--

select \* from emp where eid=201 OR ename='Alan';

select \* from emp where eid=201 OR ename='Ana';

select \* from emp where eid=201 OR ename='Jose';

3) NOT

--------

It will return the records except the condition.

A <> symbol denoted as not operator.

ex:

select \* from emp where NOT eid=202;

select \* from emp where eid<>202;

select \* from emp where job<>'Manager';

Interview Queries

----------------

Q) Write a query to display employees information whose salary is greater then

20000 and less then 50000?

select \* from emp where esal>20000 AND esal<50000;

Q) Write a query to display employees information those who are not working in 10

department?

select \* from emp where deptno<>10;

Q) Write a query to display employee information whose employee id is 201,202 and 203?

select \* from emp where eid=201 OR eid=202 OR eid=203;

Between operator

===============

Between operator returns the records those who are in the range of values.

In between operator we will take lower limit then higher limit.

ex:

select \* from student where sno between 101 and 105;

select \* from emp where esal between 5000 AND 20000;

select \* from emp where deptno between 10 AND 30;

IN operator

==============

IN operator is a replacement of OR operator.

In operator returns the records those who are matching in the list of values.

ex:

select \* from student where sno IN(101,102,103);

select \* from student where sname IN('raja','ravi','Alan');

Interview Queries

------------------

Q) Write a query to delete employees information whose employee id 201,202 and 203?

delete from emp where eid IN (201,202,203);

Q) Write a query to display employees information whose deptno number between 10 to 30?

select \* from emp where deptno between 10 and 30;

Pattern Matching operators

===========================

Pattern matching operators are used to select the letters from table.

Pattern matching operators take the support of like keyword.

We have two type of pattern matching operators.

1) Percentage (%)

2) Underscore (\_)

1) Percentage (%)

-----------------

Q) Write a query to display employees information whose emplyoee name starts with 'A' letter?

select \* from emp where ename like 'A%';

Q) Write a query to display employees information whose employee name ends with 'n' letter?

select \* from emp where ename like '%n';

Q) Write a query to display employees information whose employee name having middle letter

as 'l' letter?

select \* from emp where ename like '%l%';

2) Underscore (\_)

-----------------

Q) Write a query to display employee information whose employee name having second

letter as 'l'?

ex:

select \* from emp where ename like '\_l%';

Q) Write a query to display employee information whose employee name having second

last letter as 's'?

select \* from emp where ename like '%s\_';

Q) WRite a query to display employee information whose employee name having third

letter as 'l' ?

select \* from emp where ename like '\_\_l%';

(DML- insert , update ,delete) completed merge will be later.

DDL commands

=============

1) create (tables)

2) alter (columns)

3) drop (tables)

4) truncate (records)

5) rename (tables)

drop command

==========

It is used to drop the table from database.

syntax:

drop table <table\_name>;

ex:

drop table student;

drop table dept;

drop table emp;

rename command

================

It is used to rename the table name.

syntax:

rename <old\_name> to <new\_name>;

ex:

rename student to students;

rename emp to employees;

rename dept to departments;

truncate command

===============

It is used to delete the records permanently.

syntax:

truncate table <table\_name>;

ex:

truncate table student;

truncate table emp;

truncate table dept;

Q) What is the difference between delete and truncate command?

delete truncate

-------- ----------

It deletes the rocords temperory. It delete the records permanently.

We can rollback the data. We can't rollback the data.

Where clause can be used. Where clause can't be used.

Class 4

Alter command

=============

Using alter command we can perform following activities very easily.

i) Adding new columns

ii) Modifying the columns

iii) Renaming the columns

iv) Dropping the columns

i) Adding new columns

---------------------

Using alter command we can add new columns in a existing table.

syntax:

-----

alter table <table\_name> add (col\_name datatype(size));

ex:

alter table student ADD (state varchar2(10));

alter table student ADD (pincode number(8));

alter table student ADD (state varchar2(10),pincode number(8));

update student set state='Telangana' where sno=101;

ii) Modifying the columns

------------------------

Using alter command we can modify the columns.

We can increase or decrease the size of a column only when existing values are fit into new size.

syntax:

----

alter table <table\_name> MODIFY (col datatype(size));

ex:

desc student;

alter table student MODIFY (state varchar2(15));

desc student;

We can change the datatype of a column only when that column is empty.

ex:

desc student;

alter table student MODIFY (pincode varchar2(8));

desc student;

iii) Renaming the columns

----------------------

Using alter command we can rename the column name.

syntax:

-------

alter table <table\_name> rename column <old\_name> to <new\_name>;

ex:

alter table student rename column state to district;

alter table emp rename column esal to dailywages;

alter table emp rename column job to designation;

iv) Dropping the columns

----------------------------

Using alter command we can drop the columns.

syntax:

alter table <table\_name> drop (col);

ex:

alter table student drop (state,pincode);

alter table student drop (district,pincode);

Copy of a table or duplicate table

==================================

Using create and select command we can create copy of a table or duplicate table.

ex:

create table employees as select \* from emp;

create table employees as select \* from emp where deptno=10;

create table employee as select eid,ename,esal from emp;

create table employee as select \* from emp where eid IN (201,202,203);

create table employee as select \* from emp where comm is null;

create table employee as select \* from emp where ename like 'A%';

create table employee as select \* from emp where eid=201 and ename='Alan';

create table employee as select \* fro emp where eid<>202;

cl scr

======

It is used to clear the output screen of SQL command prompt.

syntax:

------

cl scr

Functions

==========

Functions are used to manipulate the data and give the result.

We have two types of functions.

1) Group Functions / Multiple row Functions

2) Scalar Functions / Single row Functions

1) Group Functions

-------------------

Group functions are applicable for multiple rows.

We have following list of group functions.

ex:

sum(), avg(), max(), min(), count(\*) and count(expression).

Q) Write a query to display sum of salary of each employee?

select sum(esal) from emp;

Q) Write a query to display average salary of each employee?

select avg(esal) from emp;

Q) Write a query to display highest salary from emp table?

select max(esal) from emp;

Q) Write a query to display least salary from emp table?

select min(esal) from emp;

Q) What is the difference between count(\*) and count(expression)?

count(\*)

--------

It will return number of records present in a database table.

It will return null records.

ex:

--

select count(\*) from emp;

count(expression)

-------------------

It will return number of values present in a column.

It will not include null values.

ex:

--

select count(eid) from emp; // 6

select count(comm) from emp; // 5

userlist table

==============

drop table userlist;

create table userlist(uname varchar2(10),pwd varchar2(10));

insert into userlist values('raja','rani');

insert into userlist values('king','kingdom');

commit;

Q) Write a query to check given username and password valid or not?

select count(\*) from userlist where uname='raja' AND pwd='rani'; //1

select count(\*) from userlist where uname='raja' AND pwd='rani2'; //0

Dual table

===========

Dual table is a dummy table which is used to perform arithmetic operations and to see the

current system date.

Dual table contains one row and one clumn.

ex:

select 10+20 from dual;

select 10\*5 from dual;

select sysdate from dual;

select current\_date from dual;

2)Scalar Functions

==================

We have following list of scalar functions.

i) Character functions

ii) Number functions

iii) Date functions

iv) Conversion functions

i) Character functions

----------------------

We have following list of character functions.

upper()

------

It is used to convert the string to uppercase.

ex:

select upper('oracle training') from dual;

lower()

---------

It is used to convert the string to lowercase.

ex:

select lower('ORACLE TRAINING') from dual;

initcap()

----------

It is used to display the string with initial capital letter.

ex:

select initcap('oracle training') from dual;

lpad()

-------

It is used to pad the characters to left side.

ex:

select lpad('oracle',10,'z') from dual; //zzzzoracle

rpad()

-------

It is used to pad the characters to right side.

ex:

select rpad('oracle',10,'z') from dual; //oraclezzzz

ltrim()

--------

It is used to trim the characters from left side.

ex:

select ltrim('zzoraclezz','z') from dual;// oraclezz

rtrim()

-------

It is used to trim the characters from right side.

ex:

select rtrim('zzoraclezz','z') from dual;// zzoracle

trim()

----

It is used to trim the characters from both the sides.

ex:

select trim('z' from 'zzoraclezz') from dual;

concat()

-------

It is used to concatinate two strings.

ex:

select concat('mega','star') from dual;

select concat(concat('mega','star'),'chiru') from dual;

replace()

-------

It is used to replace the characters.

ex:

select replace('gOOgle','O','oo') from dual;

ii) Number functions

---------------------

We have following list of number functions.

abs()

-----

It returns absolute value.

ex:

select abs(-45) from dual; //45

sqrt()

-----

It returns square root value.

ex:

select sqrt(25) from dual; // 5

power(A,B)

----------

It returns power value.

ex:

select power(5,3) from dual; //125

greatest()

----------

It return greatest value.

ex:

select greatest(6,9,2,4) from dual; // 9

least()

------

It returns least value.

ex:

select least(6,9,2,4) from dual; //2

ceil()

-----

It returns ceil value.

ex:

select ceil(10.9) from dual; // 11

select ceil(10.2) from dual; // 11

floor()

------

It returns floor value.

ex:

select floor(9.8) from dual; // 9

select floor(9.1) from dual; // 9

round()

-----

It returns nearest value.

ex:

select round(10.5) from dual; //11

select round(10.4) from dual; // 10

trunc()

-----------

It removes decimal numbers.

ex:

select trunc(10.56) from dual; // 10

Working with Date values

=======================

Every database software support date values.

Every database software support date values in different date patterns.

ex:

oracle - dd-MMM-yy

mysql - yyyy-MM-dd

emp1 table

----------

drop table emp1;

create table emp1(eid number(3),ename varchar2(10),edoj date);

insert into emp1 values(101,'Alan','01-JAN-24');

insert into emp1 values(102,'Jose',sysdate);

insert into emp1 values(103,'Lisa',current\_date);

commit;

Assignment

===========

Q) Write a java program to find out lucky number?

Oracle 5

iii) Date functions

--------------------

We have following list of date functions.

ADD\_MONTHS()

-------------

It is used to add the months in a given date.

ex:

select ADD\_MONTHS('07-OCT-24',4) from dual;

MONTHS\_BETWEEN()

--------------

It returns number of months between two dates.

ex:

select MONTHS\_BETWEEN('01-JAN-24','07-OCT-24') from dual;

select MONTHS\_BETWEEN('01-JAN-24','01-NOV-24') from dual;

select ABS(MONTHS\_BETWEEN('01-JAN-24','01-NOV-24')) from dual;

NEXT\_DAY()

--------

It will return next date of a given day in a week.

ex:

select NEXT\_DAY(sysdate,'sunday') from dual;

select NEXT\_DAY(sysdate,'monday') from dual;

LAST\_DAY()

------------

It will return last date of a month.

ex:

select LAST\_DAY('13-FEB-24') from dual;

select LAST\_DAY(sysdate) from dual;

iv) convertion functions

--------------------------

convertion function is used to convert from one datatype to another datatype.

ex:

TO\_CHAR() function

We have two pseduo's for TO\_CHAR

1) number TO\_CHAR

----------------

It takes '9' in digits and dollar or euro's symbol.

ex:

select eid,ename,esal from emp;

select eid,ename,TO\_CHAR(esal,'9,999') from emp;

select eid,ename,TO\_CHAR(esal,'99,999') from emp;

select eid,ename,TO\_CHAR(esal,'$99,999') from emp;

select eid,ename,TO\_CHAR(esal,'$99,999') as ESAL from emp;

2) date TO\_CHAR

---------------

select TO\_CHAR(sysdate,'dd-MM-yyyy') from dual;

select TO\_CHAR(sysdate,'yyyy-MM-dd') from dual;

select TO\_CHAR(sysdate,'HH:MI:SS') from dual;

select TO\_CHAR(sysdate,'dd-MM-yyyy HH:MI:SS') from dual;

select TO\_CHAR(sysdate,'year') from dual;

select TO\_CHAR(sysdate,'month') from dual;

select TO\_CHAR(sysdate,'day') from dual;

Integrity Constraints

=====================

Constraints are the rules which are applied on the tables to achieve accuracy and quality of data.

We have five types of contraints.

1) NOT NULL

2) UNIQUE

3) PRIMARY KEY

4) FOREIGN KEY

5) CHECK

Constraints can be created at two levels.

i) column level

ii) table level

1) NOT NULL

------------

NOT NULL constraint does not accept null values.

NOT NULL constraint can accept duplicate values.

NOT NULL constraint can be created only at column level.

column level

-------------

drop table student;

create table student(sno number(3) NOT NULL,sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(null,'ravi','delhi'); //invalid

insert into student values(101,'ravi','delhi');

Note:

-----

NOT NULL constraint can be created for multiple columns.

ex:

---

drop table student;

create table student(sno number(3) NOT NULL,

sname varchar2(10) NOT NULL,

sadd varchar2(12) NOT NULL);

insert into student values(null,'raja','hyd'); //invalid

insert into student values(102,null,'delhi'); //invalid

insert into student values(103,'ravi',null); //invalid

insert into student values(104,'ramana','pune'); // valid

2) UNIQUE

----------

UNQIUE constraint does not accept duplicates.

UNIQUE constraint can accept null values.

UNIQUE constraint can be created at column level and table level.

column level

------------

drop table student;

create table student(sno number(3) UNIQUE,sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(null,'ravi','delhi');

insert into student values(101,'ravi','delhi'); //invalid

table level

------------

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12), UNIQUE(sno));

insert into student values(101,'raja','hyd');

insert into student values(null,'ravi','delhi');

insert into student values(101,'ravi','delhi'); //invalid

Note:

-----

UNIQUE constraint can be applied to multiple columns.

ex:

---

drop table student;

create table student(sno number(3) UNIQUE,sname varchar2(10) UNIQUE,sadd varchar2(12) UNIQUE);

insert into student values(101,'raja','hyd');

insert into student values(101,'ravi','delhi'); //invalid

insert into student values(102,'raja','delhi'); //invalid

insert into student values(103,'ramana','hyd'); //invalid

3) PRIMARY KEY

----------------------

Primary key is a combination of NOT NULL and UNIQUE constraint.

Primary key does not accept null values and duplicate values.

A table can have only one primary key.

Primary key constraint can be created at column level and table level.

column level

-----------

drop table student;

create table student(sno number(3) primary key,sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(null,'ravi','delhi'); //invalid

insert into student values(101,'ravi','delhi'); //invalid

table level

------------

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12), primary key(sno));

insert into student values(101,'raja','hyd');

insert into student values(null,'ravi','delhi'); //invalid

insert into student values(101,'ravi','delhi'); //invalid

4) FOREIGN KEY

----------------------

Foreign key is used to establish the relationship between two tables.

This relationship is called parent and child relationship or master and details relationship.

To establish the relationship between two tables a parent table must primary key or unique constraint and

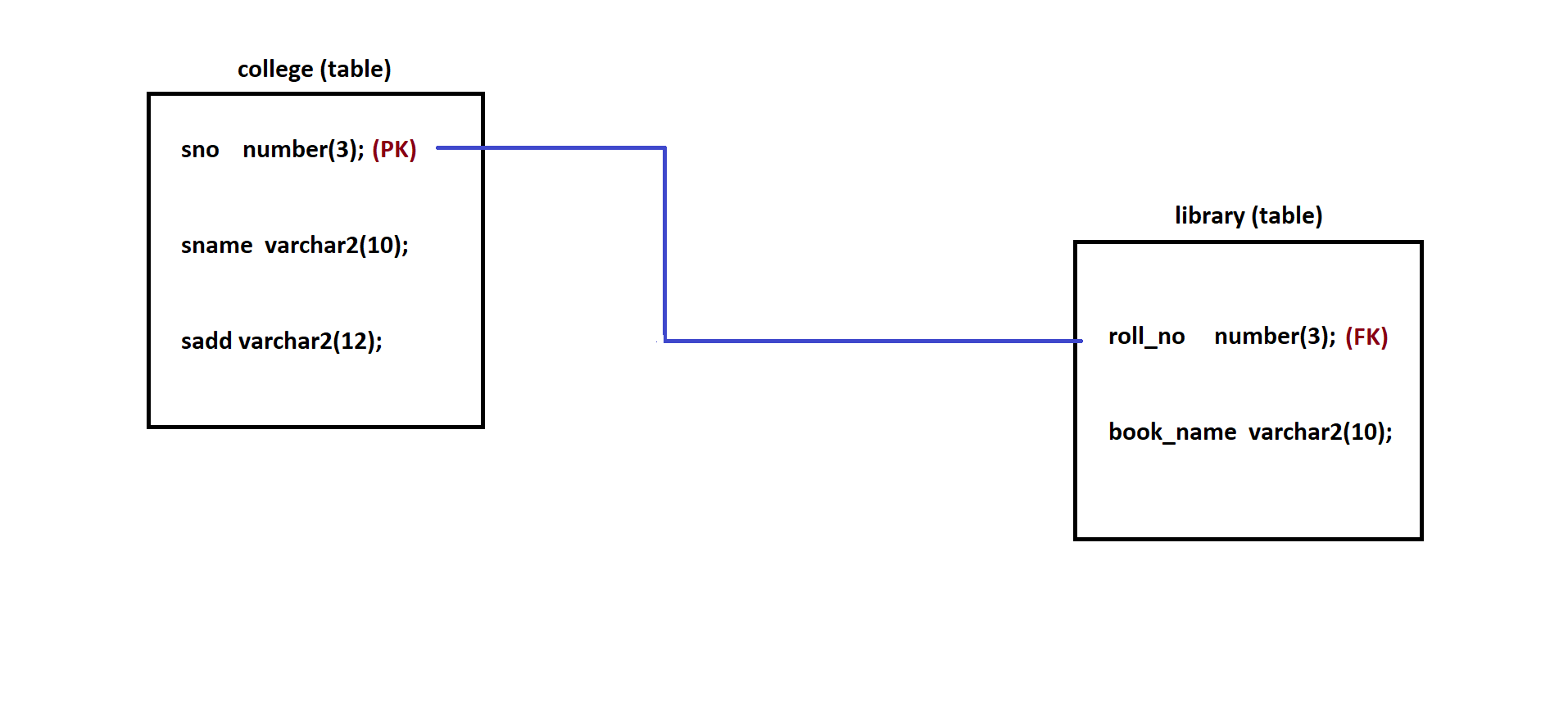
child table must have foreign key.

Foreign key will accept only those values which are present in primary key.

Foreign key name may or may not match with primary key but datatype must match.

Foreign key will accept duplicates and null values.

Diagram: oracle5.1



college table

--------------

drop table college;

create table college(sno number(3) PRIMARY KEY,sname varchar2(10),sadd varchar2(12));

insert into college values(101,'raja','hyd');

insert into college values(102,'ravi','delhi');

insert into college values(103,'ramana','vizag');

commit;

library table

-------------

drop table library;

create table library(roll\_no number(3) REFERENCES college(sno), book\_name varchar2(10));

insert into library values(101,'java');

insert into library values(102,'oracle');

insert into library values(103,'html');

insert into library values(103,'CSS');

insert into library values(null,'Spring');

insert into library values(104,'hibernate'); //invalid

In order to drop the table, first we need to drop library table then college table.

ex:

drop table library;

drop table college;

5) CHECK

---------

CHECK constraint describes domain of column.

Here domain means what type of value a column must accept.

Check contraint can be created at column level and table level.

column level

------------

drop table student;

create table student(sno number(3),sname varchar2(10),smarks number(3) check(smarks<=100));

insert into student values(101,'raja',78);

insert into student values(102,'ravi',100);

insert into student values(103,'ramana',289);

commit;

column level

------------

drop table student;

create table student(sno number(3),sname varchar2(10),smarks number(3) check(smarks between 0 and 100));

insert into student values(101,'raja',78);

insert into student values(102,'ravi',100);

insert into student values(103,'ramana',289); //invalid

commit;

column level

------------

drop table student;

create table student(sno number(3),sname varchar2(10) check (sname=lower(sname)),

smarks number(3));

insert into student values(101,'raja',78);

insert into student values(102,'RAVI',100); //invalid

insert into student values(103,'RaMaNa',289); //invalid

commit;

column level

------------

drop table student;

create table student(sno number(3),sname varchar2(10) check (sname=upper(sname)),

smarks number(3));

insert into student values(101,'raja',78); //invalid

insert into student values(102,'RAVI',100);

insert into student values(103,'RaMaNa',289); //invalid

commit;

column level

------------

drop table student;

create table student(sno number(3),sname varchar2(10),smarks number(3),

check (sname=upper(sname)));

insert into student values(101,'raja',78); //invalid

insert into student values(102,'RAVI',100);

insert into student values(103,'RaMaNa',289); //invalid

commit;

Oracle 6

Group by clause

===============

Group by clause is used to divide the rows into groups so that we can apply group functions.

A column which is used in select clause. Same column we need to use in group by clause.

Q) Write a query to perform sum of salary of each department?

select sum(esal),deptno from emp group by deptno;

Q) Write a query to display average salary of each job?

select avg(esal),job from emp group by job;

Q) Write a query to display maximum salary of each department?

select max(esal),deptno from emp group by deptno;

Q) Write a query to display minimum salary of each job?

select min(esal),job from emp group by job;

Q) Write a query to display number of employees working in each department?

select count(\*),deptno from emp group by deptno;

Having clause

=============

Having clause is used to filter the rows from group by clause.

First we need to declare group by clause then having clause.

Q) Write a query to display sum of salary of each department whose sum of salary is greater

then 35000?

select sum(esal),deptno from emp group by deptno having sum(esal)>35000;

Q) Write a query to display maximum salary of each job whose maximum salary greater then 25000?

select max(esal),job from emp group by job having max(esal)>25000;

Order by clause

===============

Order by clause is used to arrange the rows in a table.

By default it will arrange in ascending order.

ex:

select \* from emp order by eid;

select \* from emp order by ename;

select \* from emp order by esal;

select \* from emp order by esal desc;

We can write a query using where clause, group by , having clause and order by clause.

select max(esal),deptno from emp group by deptno having max(esal)>25000;

select max(esal),deptno from emp group by deptno having max(esal)>25000 order by deptno;

select max(esal),deptno from emp group by deptno having max(esal)>25000 order by deptno desc;

select max(esal),deptno from emp where deptno<>30 group by deptno

having max(esal)>25000 order by deptno ;

TCL commands

===========

1) commit

2) rollback

3) savepoint

1) commit

----------

It is used to make the changes permanent to database.

ex:

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(102,'ravi','delhi');

commit;

select \* from student; //2 records

2) rollback

-------------

It is used to undo the changes which are not permanent.

ex:

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(102,'ravi','delhi');

commit;

insert into student values(103,'ramana','delhi');

insert into student values(104,'ramulu','vizag');

select \* from student; //4 records

rollback;

select \* from student; //2 records

3) savepoint

-----------

It is used to make logical marking in database.

Instead of complete rollback we can rollback upto savepoint.

syntax:

savepoint <savepoint\_name>;

ex:

---

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

insert into student values(101,'raja','hyd');

insert into student values(102,'ravi','delhi');

savepoint sp1;

insert into student values(103,'ramana','delhi');

insert into student values(104,'ramulu','vizag');

savepoint sp2;

insert into student values(105,'Alan','Texas');

insert into student values(106,'Jose','Vegas');

select \* from student; // 6 records

rollback to sp2;

DDL commands

============

1) grant

2) revoke

Privileges

----------

Permission given to a user is called privileges.

Rights given to a user is called privileges.

We have two types of privileges.

1) System privilege : Permission given by DBA to user.

2) Object privilege : Permission given by one user to another user.

Schema

------

Schema is a memory location which is used to run SQL commands.

grant command

-----------

It is used to grant the permission to the user.

syntax:

-----

grant <privilege1>,<privilege2> to <user\_name>;

revoke command

------------

It is used to revoke the permission from the user.

syntax:

-------

revoke <privilege1>,<privilege2> from <user\_name>;

DBA> create user haritha identified by haritha;

DBA> create user rudra identified by rudra;

Rudra> conn rudra/rudra //logon denied

Haritha> conn haritha/haritha //logon denied

DBA> grant connect,resource to haritha,rudra;

Rudra> conn rudra/rudra //connected

Haritha> conn haritha/haritha //connected

Rudra>

create table employee(eid number(3),ename varchar2(10),esal number(10));

insert into employee values(201,'Kelvin',10000);

insert into employee values(202,'Jose',20000);

insert into employee values(203,'Mark',30000);

commit;

select \* from employee;

Haritha> select \* from rudra.employee; // table or view does not exist

Rudra> grant select on employee to haritha;

Haritha> select \* from rudra.employee;

Haritha> delete from rudra.employee; //insufficient privileges

rudra> grant delete on employee to haritha;

Haritha> delete from rudra.employee; //records delete

Haritha> commit;

Haritha> disc

Rudra> revoke select,delete on employee from haritha;

Rudra> disc

DBA> revoke connect,resource from haritha,rudra;

Oracle 7

Index

======

Index is an object which is used to improve the performance of select statement.

Index is a database is similar to index in a book.

Index can be created only to those columns which are widely used in where clause.

When we create index, two columns will be generated i.e One is ROWID and second is indexed column

All records will store in ascending in a indexed column.

ex:

INDEX TABLE

--------------------------------------

ROWID INDEXED COLUMN

--------------------------------------

- | 9000

- | 19000

- | 23000

- | 27000

- | 31000

- | 37000

- | 42000

---------------------------------------

We have two types of indexes.

1) Simple index

2) Complex index

1) Simple index

----------------

If index is created only for one column is called simple index.

ex:

create index idx1 on emp(esal);

Here index is used if we use esal in where condition.

ex:

select \* from emp where esal=27000;

2) Complex index

-----------------

If index is created for multiple columns is called complex index.

ex:

create index idx2 on emp(eid,deptno);

Here index is used if we use eid and deptno in where condition.

ex:

select \* from emp where eid=201 and deptno=10;

Q) Write a query to display list of indexes present in database?

select index\_name from user\_indexes;

Q) Write a query to drop the index?

drop index idx1;

drop index idx2;

Q) What is the difference between ROWNUM and ROWID?

ROWNUM

------

ROWNUM value starts with '1' and it is increment by 1.

ROWNUM values are temperory.

ex:

select eid,ename,esal from emp;

select rownum,eid,ename,esal from emp;

select rownum,deptno,dname,dloc from dept;

ROWID

------

ROWID is a memory address where our records will store in a database.

ROWID is permanent.

ex:

select rowid,rownum,eid,ename,esal from emp;

select rowid,rownum,deptno,dname,dloc from dept;

Q) Write a query to display first three records from employee table?

select \* from emp where rownum<=3;

Q) Write a query to display 4th record from employee table?

select \* from emp where rownum<=4

minus

select \* from emp where rownum<=3;

Q) Write a query to display 5th record from employee table?

select \* from emp where rownum<=5

minus

select \* from emp where rownum<=4;

Synonyms

========

Alternate name given to a table is called synonym.

It provides an alternative name for another database object, referred to as the base object, that can exist on a local or remote server.

syntax:

------

create synonym <synonym\_name> for <table\_name>;

ex:

create synonym sy1 for emp;

We can use synonym instead of table name for all commands.

ex:

select \* from sy1;

delete from sy1;

select \* from emp; // no rows selected

rollback;

Q) Write a query to display list of synonyms present in database?

select synonym\_name from user\_synonyms;

Q) Write a query to drop the synonym?

drop synonym sy1;

Views

=====

View is a virtual representation of a data from one or more then one table.

A table which is used to create a view is called base table or above table.

View does not consumes memory because it does not contain data.

View gets the data from base table when we run select command.

We have following list of views.

1) Simple view

2) Complex view

3) With read only view

4) With check option view

5) Materialized view

1) Simple view

--------------

If a view is created by using one base table is called simple view.

ex:

create view v1 as select \* from emp;

select \* from v1; // 7 records

DML operations are allowed in simple view.

ex:

delete from v1;

select \* from v1; //no rows selected

select \* from emp; // no rows selected

2) Complex view

---------------

If a view is created by using more then one table is called complex view.

ex:

create view v2 as select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e,dept d;

DML operations are not allowed for complex view.

ex:

delete from v2; // cannot delete from view

3) With read only view

--------------------

If a view is created by using one base table and DML operations are not required then we need to use

with read only view.

ex:

create view v3 as select \* from emp with read only;

select \* from v3; // 7 records

delete from v3; //cannot delete from view

4) With check option view

---------------------

If a view is created by using one base table and DML operations are allowed only when condition is true then

we need to use with check option view.

ex:

create view v4 as select \* from emp where deptno=30 with check option;

select \* from v4;

insert into v4 values(207,'Lara',40000,40,'Salesman',200); // view WITH CHECK OPTION

insert into v4 values(207,'Lara',40000,30,'Salesman',200);

select \* from v4;

select \* from emp;

5) Materialized view

--------------------

Materialized view is also known as snapshot.

We can create materialized view when table have primary key or unique key.

ex:

alter table emp ADD primary key(eid);

create materialized view v5 as select \* from emp;

select \* from v5; // 7 records

select \* from emp; // 7 records

delete from emp where eid=207;

commit;

select \* from emp; // 6 records

select \* from v5; // 7 records

We need to refresh the materialized view.

ex:

exec DBMS\_SNAPSHOT.REFRESH('V5');

select \* from v5; // 6 records

Q) Write a query to see the list of views present in a database?

select view\_name from user\_views;

Q) Write a query to drop the view?

drop view v1;

drop view v2;

drop view v3;

drop view v4;

drop materialized view v5;

Oracle 8

Sequence

========

Sequence is an object which is used to generate the numbers.

syntax:

------

create sequence <sequence\_name> start with value increment by value;

ex:

create sequence sq1 start with 1 increment by 1;

create sequence sq2 start with 101 increment by 1;

create sequence sq3 start with 10 increment by 10;

A sequence having two pseudo's.

1) NEXTVAL

--------

It is used to generate next number in a sequence.

ex:

drop table student;

create table student(sno number(3),sname varchar2(10),sadd varchar2(12));

create sequence sq1 start with 101 increment by 1;

insert into student values(sq1.NEXTVAL,'raja','hyd');

insert into student values(sq1.NEXTVAL,'ravi','delhi');

insert into student values(sq1.NEXTVAL,'ramana','vizag');

commit;

select \* from student;

2) CURRVAL

----------

It will return the last number generated by the sequence.

ex:

select sq1.CURRVAL from dual;

Q) Write a query to display list of sequences present in database?

select sequence\_name from user\_sequences;

Q) Write a query to drop the sequence?

drop sequence sq1;

Joins

======

Joins is used to retrieve the data from one or more then one table.

ex:

select \* from emp,dept; // 6\*4 = 24 records

select eid,ename,esal,deptno,dname,dloc from emp,dept; --column ambiguously defined

To overcome above limitation we need to use table.column name.

ex:

select emp.eid,emp.ename,emp.esal,dept.deptno,dept.dname,dept.dloc from emp , dept; --6\*4=24 records

Table alias

----------

A userdefined name given to a table is called table alias.

Using table alias length of the query will reduce mean while performance is maintained.

Table alias is temperory.Once the query is executed we will loss the table alias.

ex:

select e.eid,e.ename,e.esal,d.deptno,d.dname,d.dloc from emp e, dept d; -- 6\*4 = 24 records

We have following list of joins.

1) Equi Join

2) Non-Equi Join

3) Self Join

4) Cartisian Product

5) Inner Join

6) Outer Join

1) Equi Join

--------------

When two tables are joined based on common column is called equi join.

It uses the equality operator (=) to match records.

ex:

---

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e,dept d

where (e.deptno=d.deptno); -- 6 records display

2) Non-Equi Join

----------------

When table tables joined without any equi join condition is called non-equi join.

Non-equi joins are often used in scenarios where you want to join tables based on ranges of values rather than exact matches.

ex:

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e,dept d

where e.esal between 30000 AND 50000; -- 2\*4 = 8 records

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e,dept d

where e.esal>30000 AND e.esal<50000;

3) Self Join

-------------

A table join to itself is called self join.

In self join we need to create two table alias for same table.

ex:

select e1.eid,e1.ename,e1.esal,e2.job,e2.comm from emp e1,emp e2

where (e1.deptno=e2.deptno); -- 6 + 6 = 12 records

4) Cartisian Product

--------------------

If two tables are join without any condition is called cartisian product.

It returns all possible combinations.

ex:

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e,dept d; -- 6 \* 4 = 24 records

5) Inner Join

-------------

It is similar to equi join.

It is given by ANSI people.

ANSI stands for American National Standard Institute.

ex:

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e INNER JOIN dept d

ON (e.deptno=d.deptno); -- 6 records

select e.eid,e.ename,e.esal,d.dname,d.dloc from emp e JOIN dept d

ON (e.deptno=d.deptno); -- 6 records

6) Outer Join

-------------

It is a extension of equi join.

A '+' is a outer join operator.

It returns matching as well as not matching records.

We have following list of outer joins.

i) left outer join

-----------------

SQL

----

select e.eid,e.ename,e.esal,e.deptno,d.deptno,d.dname,d.dloc from emp e,dept d

where(e.deptno=d.deptno(+));

ANSI

-----

select e.eid,e.ename,e.esal,e.deptno,d.deptno,d.dname,d.dloc from emp e LEFT OUTER JOIN dept d

ON(e.deptno=d.deptno);

ii) right outer join

-------------------

SQL

----

select e.eid,e.ename,e.esal,e.deptno,d.deptno,d.dname,d.dloc from emp e,dept d

where(e.deptno(+)=d.deptno);

ANSI

-----

select e.eid,e.ename,e.esal,e.deptno,d.deptno,d.dname,d.dloc from emp e RIGHT OUTER JOIN dept d

ON(e.deptno=d.deptno);

iii) full outer join

--------------------

ANSI

-----

select e.eid,e.ename,e.esal,e.deptno,d.deptno,d.dname,d.dloc from emp e FULL OUTER JOIN dept d

ON(e.deptno=d.deptno);

Oracle 9

Merge command

=============

Merge command is possible by using update and insert command.

ex:

student10 table

---------------

drop table student10;

create table student10(sno number(3),sname varchar2(10),sadd varchar2(12));

insert into student10 values(101,'raja','hyd');

insert into student10 values(102,'ravi','delhi');

insert into student10 values(103,'ramana','vizag');

commit;

student20 table

---------------

drop table student20;

create table student20(sno number(3),sname varchar2(10),sadd varchar2(12));

insert into student20 values(103,'Alan','Florida');

insert into student20 values(104,'John','Texas');

commit;

merge into student10 s1

using student20 s2

ON(s1.sno=s2.sno)

when matched then

update set sname=s2.sname,sadd=s2.sadd

when not matched then

insert(sno,sname,sadd) values(s2.sno,s2.sname,s2.sadd);

select \* from student10;

select \* from student20;

Sub Queries

============

We will declare a query inside another query such concept is called sub query.

In sub query , first inner query will execute then outer query.

Sub queries are used to select the records based on unknown values.

We have following list of sub queries.

1) Single Row subquery

2) Multiple Row subquery

3) Multiple Column subquery

1) Single Row subquery

-----------------------

If a sub query returns only one row is called single row sub query.

Sub queries can be nested upto 32 levels.

ex:

SQL

----

select \* from emp where eid=201;

Subquery

--------

select \* from emp where eid=(select eid from emp where ename='Alan');

ex:

SQL

---

select \* from emp where eid=201 and ename='Alan';

Subquery

---------

select \* from emp where eid=(select eid from emp where ename='Alan')

and ename=(select ename from emp where esal=9000);

Q) Write a query to display employees information whose salary is greater then mark salary?

select \* from emp where esal>(select esal from emp where ename='Mark');

Q) Write a query to display second highest salary from emp table?

select max(esal) from emp where esal<(select max(esal) from emp);

2) Multiple Row subquery

---------------------

If a sub query returns more then one row is called multiple row sub query.

To perform multiple row sub query we need to use multiple row operators.

We have three multiple row operators.

i) ANY

ii) ALL

iii) IN

i) ANY

-----

select \* from emp where esal > ANY (select esal from emp where deptno=10);

select \* from emp where esal < ANY (select esal from emp where deptno=10);

ii) ALL

------

select \* from emp where esal > ALL (select esal from emp where deptno=10);

select \* from emp where esal < ALL (select esal from emp where deptno=10);

iii) IN

---------

select \* from emp where esal IN (select esal from emp where deptno=10);

3) Multiple Column subquery

--------------------------

If a sub query returns more then one column is called multiple column subquery.

ex:

select \* from emp where(eid,ename,esal) IN (select eid,ename,esal from emp where eid=201);

select eid,ename,esal from emp where(eid,ename,esal) IN (select eid,ename,esal from emp where eid=201);

select eid,ename,esal from emp where(eid,ename,esal) IN (select eid,ename,esal from emp);

Assignment

===========

Q) Write a query to display last three records from emp table?

SELECT \* FROM ( SELECT \* FROM emp ORDER BY eid DESC) WHERE ROWNUM <= 3 order by eid;

Q) What is the difference between RDBMS and MongoDB?

RDBMS MongoDB

------------- -----------

It is a relational database. It is a non-relational or document based database.

It can't store the data in key and value pair. It stores the data in key and value pair.

Not suitable for hierarchical data storage. Suitable for hierarchical data storage.

It has a predefined(static) schema. It has a dynamic schema.

It contains tables. It contains collections.

It is row based. It is document based.

It is column based. It is field based.

It is slower. It is faster.

It supports SQL query language. It supports JSON query language.

PL/SQL

=======

PL/SQL stands for Procedural Language extension to Structure Query Language.

It is a extension of SQL and gives following features.

1) We can achieve programming features like control statements ,loops and etc.

2) It reduces network traffic.

3) We can display custom error messages by using the concept of exception handling.

4) We can perform related operations by using the concept of triggers.

5) We can save the source code permanently to database for repeated execution.

PL/SQL block

============

A PL/SQL program is also known as PL/SQL block.

syntax:

DECLARE

-

- -- Declaration Section

-

BEGIN

-

- -- Execution Section

-

EXCEPTION

-

- -- Exception Section

-

END

/

Here '/' is used to submit the PL/sQL block into a database.