**MONGO\_DB**

MongoShell::

it is a javascript shell that allows to interact with mongodb instanc in command line.

that one can perform administrative functions inspecting and instance of mongodb.

to enter the ongo shell there is a command line which is mongosh/mongo.

mongodb stores data in JSON format

databases

mySql,postgres firebase(google),Redis,mongodb

stores data in tabular stores data in key object notation

form

Database:

we will interact with server using http(hyper text transfer protocol) modules

http is used to create a local server

difference of structure of mysql & mongodb

mySql mongodb

===================================|

| nAME |YEAR OF MANUFACTURE | [{NAME:BMW,YEAR:1916},

|==================================| {NAME:FORD,YEAR:1903}

| BMW | 1916 | {...}

|==================================| ]

| HONDA | 1903 |

|==================================|

mongodb IS easy to learn &fast access compared to mysql

mysql has limitted storage as it is stores data tabular form

there are inconsistencies in mysql.

there are fields in mongodb.

group of fileds is known as documents.

group of documnets known as collection.

group of collection known as Database

user

/ \

/ \

/ \

mongodb---------ODM(object document mapper)

DIFFERENCE BETWEEN MONGODB AND RDBMS

MONGODB RDBMS

it is a non relational database | it is a relational database

it is suitable for hierarical data storage | it is not suitable

it is a dynamic schema | it was an predefined schema

it is faster in terms of performance than | it slower in terms of performance than

RDBMS | MONGODB

Features of mongodb

1)schema LESS::one collecions there no need that all documments should contain same fields

2)document oriented

3)Indexing

4)scalability:mongodb provides horizontal scalability with help of sharding.

sharding means maintaining more than one database copy to share with server

5)Replication::To prevent dataloss after the servercrash

if server is failed data is fetched from other server to user.this happens because

data in server replicated in different servers.

6)Aggregation::it allows to perform ooperaions on a group of data and get a single result.

7)High Performance::because of other features mongodb performance is high

DISADVANTAGES

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

\*it uses high memory for data storage.

\*not allow to store more than 16MB data.

\*Nesting of data in BSON is also limitted your are not allowed to nest data

more than 100 levels.

MONGODB queries

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

MONGODB OPERATORS ||

EQUALITY:::$EG

LESS THAN:::$LT

LESS THAN EQUAL TO::::$LTE

GREATER THAN:::::$GT

GREATER THAN EQUAL::::$GTE

NOT EQUALS::::::::$NE

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

TO USE/CREATE A DATABASE

::::: use DATABASE\_NAME

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

TO CREATE A COLLECTION

::::::db.createCollection("collection\_name");

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

TO VIEW THE COLLECTION IN THE DATABASE:

show collecions

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

TO VIEW THE CURRENT WORKING DATABASE

show db

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

TO VIEW WHETHER COLLECTION IS IN DATABASE

db.collectionname

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

TO DISPLAY ALL THE CONTENT IN COLLECTION OF A DATABASE::

WE USE find() IN DIFFERENT CASES.

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

TO DISPLAY ALL THE CONTENT

::::db.COLLECTION\_NAME.find();

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

TO DISPLAY ONLY ONE DOCUMENT

::::db.COLLECTION\_NAME.findOne({key:value});

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

TO DISPLAY ONLY THE DOCUMENTS WHICH CONTAINS GIVEN SPECIFIC CONTENT

:::db.COLLECTION\_NAME.find({key:value});

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

TO DISPLAY ONLY THE DOCUMNETS WHICH SATISFIES ALL THE GIVEN CONDITIONS::

db.COLLECTION\_NAME.find({$and:[{key1:val1},{key2:val2},{key3:val3}]});

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

TO DISPLAY ONLY THE DOCUMENTS WHICH SATISIFES ANY ONE OF GIVEN CONDITIONS:::

db.COLLECTION\_NAME.find({$or:[{key1:val1},{key2:val2},{key3:val3}]});

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

TO DISPLAY THE DOCUMENT WHICH DOES NOT SATISFY THE GIVEN CONDITION:

db.COLLECTION\_NAME.find({$not:[{KEY1:VAL1},{KEY2:VAL2}]});

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

TO DISPLAY ONLY SPECIFIC FILEDS IN THE DOCUMENT::

db.COLLECTION\_NAME.find({},{field:1/0,FILED2:1/0,FIELD:1/0,...});

1====TO DISPLAY FIELD.

0====NOT TO DISPLAY FIELD.

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

TO DISPLAY ONLY A SPECIFIC NUMMBER OF DOCUMENTS:::

db.COLLECTION\_NAME.find({key:value}).limit(number of documnets);

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

TO know number of documents in a collection

db.COLLECTION\_NAME.find().count();

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

TO display n number of documents and skip last k documnets

db.COLLECTION\_NAME.find().limit(n).skip(k);

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

TO display DOCUMENTs in sorted order according to key

db.COLLECTION\_NAME.find().sort({KEYname:1});

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

TO FIND THE DOCUMENTS WHICH CONTAINS ANY ONE OF THE VALUE IN GIVEN VALUES

db.COLLECTION\_NAME.find({ field: { $in: [<value1>, <value2>, ... <valueN> ] } })

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

TO FIND THE DOCUMENTS WHICH NOT CONTAINS ANY ONE OF THE VALUE IN GIVEN VALUES

db.COLLECTION\_NAME.find({ field: { $nin: [<value1>, <value2>, ... <valueN> ] } })

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

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

INSERTING DATA INTO DATABASE

TO INSERT ONE DOCUMENT INTO COLLECTIONS

db.COLLECTION\_NAME.insertOne({field1:value,filed2:value,filed3:value,...});

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

TO INSERT MORE THAN ONE DOCUMENTS INTO COLLECTIONS

db.COLLECTION\_NAME.insertMany([{field1:value,filed2:value,filed3:value},{field1:value,filed2:value,filed3:value},{field1:value,filed2:value,filed3:value},...]);

or

db.COLLECTION\_NAME.insert([{field1:value,filed2:value,filed3:value},{field1:value,filed2:value,filed3:value},{field1:value,filed2:value,filed3:value},...]);

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

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

UPDATE DOCUMENTS IN DATABASE

TO UPDATE DATA OF A SINGLE FILE.

db.COLLECTION\_NAME.updateOne({key:value},$set:{newkey:value});

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

TO UPDATE DATA OF MULTIPLE FILE IN DATABASE

db.COLLECTION\_NAME.update({KEY:VALUE},{$set:{FILED:NEWVALUE},{....}},{multi:true})

or

db.COLLECTION\_NAME.updateMany({KEY:VALUE},{$set:{FILED:NEWVALUE},{....}},{multi:true})

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

to access document in based on the value in an object/array.

db.COLLECTION\_NAME.find({"object.name":"value"})

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

TO DELETE DATA FORM DATABASE

TO DELETE A SINGLE FILE FROM COLLECTION.

db.COLLECTION\_NAME.deleteOne({key:value});

TO DELETE MULTIPLE FILES FROM COLLECTION.

db.COLLECTION\_NAME.deleteMany({key:value});

or

db.COLLECTION\_NAME.remove({key:value});

TO DELETE ALL THE DATA IN THE COLLECTION.

db.COLLECTION\_NAME.remove({});

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

TO DELETE THE COLLECTION FORM DATABASE.

db.COLLECTION\_NAME.drop();

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

TO DELETE THE ENTIRE DATABASE

db.dropDatabase();

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

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

CAP theorem:

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

C->Consistency:::All the node in the network see the same data at a same time or reader get the most recently written data.

A->Availability:::if the server is not crashed it has to process the user request/it means server must fulfill the request

if it is not crashed,which means if the sever is not crashed and user makes the call to the node to node cannot ingonre the request.

P->Partition Tolerance

None of any two combinations of the given three are not gonna work at same time.

Eric Brewers theorem:: states that atmost two the given combinations can work togather in CAP

gave the CAP theorem.

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

Distributed System

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

it is anetw0ork of autonomous computer thatworks togather as a single system to perform a

specific task/set specific tasks.

Inconsistency:

this is what means that all the nodes of the distributed system should have consistence data.

**SQL**

set serveroutput on verify off;

it used to display the content of oracle that work for in our screen

usage ...

structure

declare

.

. to declare the variables

.

begin

.

.to assign the values to variables.

.

procedure/executable statements.

.

.

end;

/

to print some content

dbms\_output.put\_line();

TO TAKE INPUT FROM USER

VARAIABLE\_NAME:=&VARIABLE\_NAME;

example:::

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

SQL> set serveroutput on verify off;

SQL> declare

2 a number;

3 b number;

4 begin

5 a:=9;

6 b:=13;

7 dbms\_output.put\_line('value of a= '||a);

8 end;

9 /

value of a= 9

PL/SQL procedure successfully completed.

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

using conditional statements in PL/SQL

if condition then

statements ......

elsif condition then

statements ......

else

statements......

SQL> declare

2 a number;

3 b number;

4 c number;

5 begin

6 a:=10;

7 b:=20;

8 c:=30;

9 if a>b then

10 if a>c then

11 dbms\_output.put\_line(a||'is greater among three');

12 end if;

13 end if;

14 if b>a then

15 if b>c then

16 dbms\_output.put\_line(b||'is greater among three');

17 end if;

18 end if;

19 if c>a then

20 if c>b then

21 dbms\_output.put\_line(c||'is greater among three');

22 end if;

23 end if;

24 end;

25 /

30is greater among three

PL/SQL procedure successfully completed.

SQL> declare

2 a number;

3 b number;

4 c number;

5 begin

6 a:=10;

7 b:=20;

8 c:=30;

9 if a>b then

10 if a>c then

11

12 dbms\_output.put\_line(a||'is greater among three');

13 else dbms\_output.put\_line(c ||' is greater than'||a||','||b);

14 end if;

15 elsif b>c then

16 dbms\_output.put\_line(b ||' is greater than'||a||','||c);

17 else dbms\_output.put\_line(c ||' is greater than'||a||','||b);

18 end if;

19 end;

20 /

30 is greater than10,20

PL/SQL procedure successfully completed.

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

loops

1)simple loop

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

loop

statements....

exit when condition;

end loop;

EXAMPLE:::::::::

SQL> declare

2 a number;

3 begin

4 a:=1;

5 loop

6 dbms\_output.put\_line(a);

7 a:=a+1;

8 exit when a>6;

9 end loop;

10 end;

11 /

1

2

3

4

5

6

PL/SQL procedure successfully completed.

2)while loop

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

while condition

loop

statements...

end loop;

EXAMPLE

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

SQL> declare

2 a number;

3 begin

4 a:=2;

5 while a<=10

6 loop

7 dbms\_output.put\_line(a);

8 a:=a+2;

9 end loop;

10 end;

11 /

2

4

6

8

10

PL/SQL procedure successfully completed.

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

to factorial of number

SQL> declare

2 i number;

3 n number;

4 f number;

5 begin

6 i:=1;

7 n:=6;

8 f:=1;

9 while i<=n

10 loop

11 f:=f\*i;

12 i:=i+1;

13 end loop;

14 dbms\_output.put\_line(n||' factorial = '||f);

15 end;

16 /

6 factorial = 720

PL/SQL procedure successfully completed.

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

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

PROGRAMTO FIND SUM OF FIRST N NATURAL NUMBERS.

SQL>declare

2 i number;

3 n number;

4 s number;

5 begin

6 i:=1;

7 n:=&n;

8 s:=0;

9 while i<=n

10 loop

11 s:=s+i;

12 i:=i+1;

13 end loop;

14 dbms\_output.put\_line('sum of first '||n||' numbers = '||s);

15 end;

16 /

Enter value for n: 10

sum of first 10 numbers = 55

PL/SQL procedure successfully completed.

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

3) FOR loop

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

for variable IN start..endrange

loop

statement....

end loop;

SQL> declare

2 i number;

3 n number;

4 fac number;

5 begin

6 i:=1;

7 fac:=1;

8 n:=&n;

9 for i IN 1..n

10 loop

11 fac:=fac\*i;

12 end loop;

13 dbms\_output.put\_line(n||' factorial = '||fac);

14 end;

15 /

Enter value for n: 5

5 factorial = 120

PL/SQL procedure successfully completed.

JDBC API

java.sql.DriveManager->interface

java.sql.Sqlexception->interface

TO CHECK THE DRIVER::::::;;javap com.mysql.jdbc.Driver

**SQL\_CONNECT\_JDBC**

import java.sql.\*;

import java.util.\*;

class Insert{

public static void main(String[] args) {

try{

Class.forName("com.mysql.cj.jdbc.Driver");

String url="jdbc:mysql://localhost:3306/student";

String username="root";

String password="system";

Connection con=DriverManager.getConnection(url,username,password);

String q="create table table1( name varchar(20),age int,id int not null);";

String p="insert into table1 values('hello12',19,15455);";

Statement stmp=con.createStatement();

//FOR CREATION OF TABLE USE STATEMENT::STMP.executeUpdate(p);

//TO INSERT DATA INTO THE TABLE

stmp.executeUpdate(p);

String d="update table1 set name='ew2' where id=455";

String o="select \* from table1";

stmp.executeUpdate(d);

ResultSet r=stmp.executeQuery(o);

while(r.next())

{

System.out.println(r.getString(1)+" "+r.getString(2)+" "+r.getString(3));

}

// System.out.println("table is created"+ k);

con.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

import java.sql.\*;

import java.util.\*;

class Sample{

public static void main(String[] args) {

try{

Class.forName("com.mysql.cj.jdbc.Driver");

String url="jdbc:mysql://localhost:3306/student";

String username="root";

String password="system";

Connection con=DriverManager.getConnection(url,username,password);

if(con.isClosed())

{

System.out.println("Connection closed");

}

else

{

System.out.println("Connection created..");

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

**SQL**

TO CONNECT WITH DATABASE

CONNECT

tto show the current user::::SHOW USER

TO CREATE USE

CREATE USER USERNAME

TO CHANGE THE USERNAME

alter user oldname newname;

TO change the password

alter is used

TO DROP USER

drop user username;

TO GRANT PERMISSON

grant role to username;

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

|| ||

|| TABLE ||

|| ||

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

TABLE is a DATABASE object which is used to store data in relational database

TO CRATE A TABLE

create table tablename(

fieldname1 datatype,

...

filednamen datatype

);

TO VIEW THE SCHEMA OF THE TABLE

desc tablename

TO RENAME THE TABLE NAME

rename oldname to newname;

TO ADD/REMOVE/ RENAME A SPECIFIC COLUMN TABLE WE USE alter

ex;

to addcolumn::::alter table tablename add columnname datatype;

to remove column:::alter table tablename drop column columnname;

torename column:::alter table tablename rename column oldnameto newname;

to change datatype of column::: alter table tablename modify columnaname newdatatype;

TRUNCATE is used to remove all data except the SCHEMA of Table

truncate table tablename;

DDL(data definition language) commands to define the table.

TO GRANT SPACE FOR THE USER

grant unlimited tablespace to username;

TO UPDATE THE VALUE INTO TABLE

UPDATE TABLENAME SET columnaname=VALUE WHERE CONDITON;

EX::

update instructor set name='isha' where id=123;

OR operator:

atleast one of the condition must be true for the row to be selected.

ex::select e\_name,department from emp where gender='m' or salary<50000;

AND operator:

all of the given the condition must be true for the row to be selected.

ex:::select e\_name,department from emp where gender='m' and salary>453323;

the specified condition must be false for the roq to be selected.

select \* from emp where geneder!='m';

TO SELECT Arow on a SPECIFIC PATTERN

::this is used to list all the rows in the table in a table whose column values matches specified pattern.

select \* from emp where ename like('%h');

In operator

::this is used when you want to compare the column with more than one value present in the table.

In working is similar to OR operator and ;

select e\_name,e\_salary from emp where e\_id in('201','202','205');

between operator

used to compare data for a range of values

column values is b/w twovalues including end values

select e\_id,e\_name from emp where e\_salary between 10000 and 300000;

constraints:data-constraints sql uses integrity constraints to prevent invalid data entry

into the table of database;

1)not null:applied to column level only & not to table

2)unique::applied to both column & table level

3)primary key::applied toboth column & table level

4)Foriegn key::Referential\_Integrity\_Contraints.it identify

it establish relation between two clumn in same table or different table.

foreign key is defined in child table and the table containing the reference column in the parent table.

this key can be applied in both column & table level.

systnax:::::<childcolumn> <datatype> [constraint <name>] references <parent table>[(<parent-column>)].

5)check::applied to both column & table level.check :: also knownas business\_Integrity\_rule constraints.

allows you to define a condition that a value aentered into table must satisfies before entering accepting into table.

6)default::only column level.

Main Difference between Unique & Primary Key is:

Unique can accept null values.

CLASS

SQL> desc team;

Name Null? Type

-

ID\_NO NOT NULL NUMBER(6)

TEAM\_NAME NOT NULL VARCHAR2(30)

POSITION VARCHAR2(30)

SQL> desc player;

Name Null? Type

-

ID\_NO NUMBER(6)

PLAYER\_NAME VARCHAR2(30)

HEIGHT NUMBER(3)

create table cab(

2 id number(4),

3 liscene\_plate varchar2(20),

4 ar\_model\_id number(9),

5 manufacture\_year number(15),

6 primary key(id),

7 unique(liscene\_plate));

create table driver(

2 id number(4),

3 name varchar(30),

4 dob date,

5 driving\_liscene\_no varchar2(20),

6 foreign key (id) references cab(id));

sQL joins::used to relate info in differnet tables

joins are used to combine coulmns from different tables.

types

1.equviJoin

2. CartesianJoin

3.outerJoin

a.LeftOuterJoin

b.RightOuterJoin

4.SelfJoin

EQUVIJOIN::

select table1.column,table2.column....tablen.column

table table1,table2,...tablen where table1.column=table2.column;

SQL> create table r1(

2 first\_name varchar2(10),

3 last\_name varchar2(20),

4 salary number(10));

Table created.

SQL> create table r2(

2 last\_name varchar2(20),

3 gender varchar2(10),

4 age number(2));

Table created.

SQL> select \* from r1;

FIRST\_NAME LAST\_NAME SALARY

sai kumar 15000

vasavi reddy 20000

ajay anala 300000

SQL> select \* from r2;

LAST\_NAME GENDER AGE

kumar male 22

garg female 19

anala male 19

SQL> select first\_name,salary,age from r1,r2 where r1.last\_name=r2.last\_name;

FIRST\_NAME SALARY AGE

sai 15000 22

ajay 300000 19

Cartesian joinsSQL> select first\_name,salary,age from r1,r2;

FIRST\_NAME SALARY AGE

sai 15000 22

sai 15000 19

sai 15000 19

vasavi 20000 22

vasavi 20000 19

vasavi 20000 19

ajay 300000 22

ajay 300000 19

ajay 300000 19

9 rows selected.

SQL JOIN CONDITION RETURNS ALL ROWS FROM BOTH TABLES WHICH SATISFIES THE JOIN CONDITION ALONG WITH ROWS WHICH DO NOT SATISFIES JOIN CONDITON

FORM ONE THE TABLE .IT IS IN SUCH CASES IF THERE ARE ANY VALUE SIN ONE TABLE THAT DO NOT HAVE CORRESPONDING VALUES IN THE OTHER SUCH ROWS CAN BE

FORCEFULLY SELECTED BY USING THE OUTE JOIN SYSMBOL (+) IT IS USED ON ONE SIDE OF JOIN CONDITION ONLY.

r1(+)=r2 right outer joins preference r2.

r1=r2(+) left outer join preference r1.

SQL> select first\_name,salary,age from r1,r2 where r1.last\_name=r2.last\_name(+);

FIRST\_NAME SALARY AGE

sai 15000 22

ajay 300000 19

vasavi 20000

SQL> select first\_name,salary,age from r1,r2 where r1.last\_name(+)=r2.last\_name;

FIRST\_NAME SALARY AGE

sai 15000 22

ajay 300000 19

19

SELF JOIN

SQL> select surname.last\_name as last\_name,sex.gender as gender from r2 surname,r2 sex where surname.gender=sex.gender;

LAST\_NAME GENDER

anala male

kumar male

garg female

anala male

kumar male

VIEW

SQL> create view virtual as select \* from emp where id<155;

View created.

SQL> select \* from virtual;

NAME AGE SALARY ID

david 17 16500 154

james 22 20000 104

vasavi 23 5000 100

sai 23 46000 90

SQL>