17/04/2018

PLSQL

MEGHDUT
PAGE NO. DATE / /20

nvl2 (column_name, if not null, if null)

nvl2 → select nvl2 (grade, 'pass', 'fail') from salary;

as sequence is a db object, index is the same.

index → create index index_name on table_name (col_name) ;

we need index when we have huge amount of data... this improves performance.

Secure Random Number Generator
mac address, port number, IP address, OS (32 bit / 64 bit)
these 4 things are used to generate random number.

unique index does not allow duplicate values.

unique index → create unique index index_name on table_name (col_name);

drop index → drop index index_name ;

btree is default index in oracle

show index → select * from user_indexes ;

equi join
non-equi join
inner join
self join
full outer join

equi join →
```
select e.first_name, d.department_name
from employees e, departments d
where e.department_id = d.department_id
```

using clause →
```
select e.first_name, d.department_name
From employees e join departments d
using (department_id);
```

on clause →
```
select e.first_name, d.department_name
from employees e join departments d
on (e.department_id = d.department_id);
```

non-equi join →
```
select e.first_name, d.department_name
from employees e, department d
where e.department_id > d.department_id;
```

self join →
```
select e1.employee_id, e2.employee_id
from employees e1, employees e2
where e1.department_id = e2.Manager_id
```

500 is the max. no. of records visible on screen
2889 is 107 (employees entry) x 27 (departments entry)

Full outer join →
```
select e.first_name, d.department_name
from employees e join departments d
on (e.department_id = d.department_id);
```

count →
```
select count(*) from salary;
```

sql200.net

single row subquery →
```
select * from employees
where salary >
(select salary from employees
 where employee_id = 108);
```

multi row subquery →
```
select * from employees
where salary > any
(select salary from employees
 where salary between 2500 and 4000);
```

multiple row subquery →
```
select first_name, salary, job_id from employees
where salary > any
(select avg(salary) from employees
 group by job_id);
```

multiple subquery → select * from employees
where salary =
(select max (salary)
from employees
where salary <
(select max (salary)
from employees));

2889 - Total Count
Emp - 107
depart - 27
1605

union → select * from student1
union select * from student2 ;

union all → select * from student1 union all select * from student2 ;

union all also gives duplicate results

datatype of columns should match in union and all.

select first_name, manager_id from employees union
select department_name, manager_id from departments ;

intersect → select * from stud1 intersect select * from stud2 ;

minus → select * from stud1 minus select * from stud2 ;

sum → select sum (salary) from employees ;

avg → select avg (salary) from employees ;

max → select max (salary) from employees ;

min → select min (salary) from employees ;

count → select count (salary) from employees ;

PLSql → Programming language + Sql

1. cursor
2. triggers (db) / schedulars (programming)
3. stored procedure

int a = 30; in C do { } while()
a int := 30; in plsql do: while()

cursor is used to store data of multiple datatype but array is of single datatype.

create function function_name;

cursor is used to store huge amount of data but it is only temperary just like array.

Modes :
1. in - keyboard
2. out - monitor screen
3. error -

start → set serveroutput on
~~start~~ → Begin
display dbms_output.putline('Welcome');
end;
→ Welcome

concat → begin
dbms_output.put_line('welcome' || 'Hello');
end;
→ WelcomeHello

declare → declare
num number := 20;
doj date default sysdate;
begin
--num := 25;
dbms_output.put_line('Result is' || num);
end;

```
    dbms_output.put_line ('Result is '|| doj);
end;
```

↳ Result is 20
~~Result is~~ 09-sep-18

→
```
declare
    num number:= 30;
begin
    dbms_output.put_line ('the value ');
    dbms_outpt.put_line (' is '|| num);
end;
```

↳ the value
is 30

→
```
declare
    num constant number := 80;
begin
    dbms_output.put_line ('value is '|| num);
end;
```

↳ value is 80

this const. value can not be changed

→ ~~declare~~ ~~num constant~~

```
create table plsql
( uname varchar2(20),
  sal number );
```

```
declare
    unames varchar2(20);
    sals number;
begin
    unames := '&unames';
    sals := &sals;
    insert into plsql values (unames, sals);
end;
```

↳

Enter values for unames : Srinivas
Enter value for sals : 560

select * from plsql;

↳

Uname	sal
Srinivas	560

→
```
declare
    unames varchar2(20);
    sals number;
begin
    select uname, sal
    into unames, sals
    From plsql;
    dbms_output.put_line(unames || sals);
end;
```

↳ Srinivas 560

%type is predefined attribute used for changing data type
%rowtype is also same.

var_name table_name.column_name%type

%type →
```
declare
    unames plsql.uname%type;
    sals plsql.sal%type;
begin
    dbms_output.put_line(unames || sals);
end;
```

uname	salary	roll
Abc	560	68
Def	890	69
Ghi	230	70

→ data type is called %type

→ %Rowtype

for loop →

```
begin
    for index1 in 1 .. 5
    loop
        dbms_output.put_line('test');
    end loop;
end;
```

↳

```
test
test
test
test
test
```

while loop →

```
declare
    num number := 1;
begin
    dbms_output.put_line('Value is ' || num);
    while num <= 5
    loop
        dbms_output.put_line('test');
        num := num + 1;
    end loop;
end;
```

↳

```
Value is 1
test
test
test
test
test
```

→

```
declare
    a number := 1;
    b number := 2;
begin
    while a <= 10
    loop
```

PAGE NO. DATE / /20

```
        dbms_output.put_line( b||'*'||a||'='||b*a);
        a := a+1;
    end loop;
end;
```

↳ 2*1 = 2
2*2 = 4
:
2*10 = 20

→
```
declare
    b number:= 2;
begin
    for a in 1..10
    loop
        dbms_output.put_line (b||'*'||a||'='||b*a);
    end loop;
end;
```

→
```
declare begin
    for a in 65..90
    loop
        dbms_output.put_line (chr(a));
    end loop;
end;
```

↳ A
B
C
:
Z

10/04/2018

else if →

```
declare
    did number(2);
begin
    select department_id into did
    from departments where department_name 'IT';
    if did =40 then
        dbms_output.put_line ('New York');
    elsif did = 60 then
        dbms_output.put_line ('Dallas');
    else
        dbms_output.put_line ('Boston');
    end if;
end;
```

↳ Dallas

case →

```
declare
    did number(2);
begin
    select department_id into did
    from departments where department_name = 'IT';
    case did
        when 40 then
            dbms_output.put_line ('New York');
        when 60 then
            dbms_output.put_line ('Dallas');
        else
            dbms_output.put_line ('Boston');
    end case;
end;
```

labled case →

```
declare
    did number(2);
begin
    select department_id into did
    from departments where department_name = 'IT';
    << my_case >>
    case
```

```
case did
    when 40 then
        dbms_output.put_line('New York');
    when 60 then
        dbms_output.put_line('Dallas');
end case my_case;
end;
```

searched case →

```
declare
    did number(2);
begin
    select department_id into did from departments
    where department_name = 'IT';
    case
        when did = 40 then
            dbms_output.put_line('New York');
        when did = 60 then
            dbms_output.put_line('Dallas');
    end case;
end;
```

simple loop →

```
declare
    i number := 1;
begin
    loop
        dbms_output.put_line('i = ' || i);
        i := i + 1;
    exit when i > 5;
    end loop;
end;
```

↳

```
i = 1
i = 2
i = 3
i = 4
i = 5
```

%Row Type →
```
declare
    dept salary%ROWTYPE ;
begin
    select * into dept from salary where uname='Raj';
    dbms_output.put_line (dept.uname) ;
    dbms_output.put_line (dept.salary) ;
    dbms_output.put_line (dept.grade) ;
end;
```
↳
Raj
3194
D

reverse for →
```
begin
    for i in reverse 1..5
    loop
        dbms_output.put_line ('i = '||i);
    end loop;
end;
```
↳
i = 5
i = 4
i = 3
i = 2
i = 1

goto label and Null →
```
begin
    for i in 1..50
    loop
        dbms_output.put_line ('i = '||i);
        if i = 40 then
            goto here;
        end if;
    end loop;
    <<here>>
    Null;
end;
```
↳
i = 1
i = 2
i = 3
i = 4

create view → create view V as select * from dept with read

devops → TDD
BDD

β - version is for testing

forsible view can be created

→ create or replace force view V as select * from salary ;

drop view → drop view V ;

view → select * from V ;

Cursor →
```
Declare
    CURSOR cName IS SELECT * FROM departments ;
    deptdetail department% ROWTYPE ;
begin
    OPEN cName ;
    FETCH cName INTO deptdetail ;
    CLOSE cName ;
    DBMS_OUTPUT.PUT_LINE (deptdetail.dname) ;
end ;
```

%not found →
```
declare
    CURSOR cName IS SELECT * FROM departments
    dept detail departments % ROWTYPE ;
begin
    if cName % ISOPEN
    then NULL ;
    else OPEN cName ;
    end if ;
    fetch cName INTO deptdetail ;
    loop
```

```
        fetch cname INTO deptdetail;
        exit when cname%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(deptdetail.dname);
    end loop;
end;
```

%found →

```
declare
    cursor cName is select * from departments;
    deptdetail departments%ROWTYPE;
begin
    if cname%ISOPEN then null;
    else open cname;
    end if;
    fetch cname into deptdetail;
    while cname%FOUND
    loop
        dbms_output.put_line(deptdetail.dname);
        fetch cname into deptdetail;
    end loop
    close cname;
end;
```

%rowcount →

```
if cname%ROWCOUNT > 0
then
    while cname%found
    loop
        dbms_output.put_line(deptdetail.dname);
        fetch cname into deptdetail;
    end loop;
else
    dbms_output.put_line('Data not found');
end if;
```

PAGE NO.

psendo attributes of plsql :

cursor is a composite datatype
but cursor does not have a datatype

→
```
declare
    cursor cname is select * from employees where
    deptno = & no ;
    empdetail employees % rowtype ;
begin
    if cname % isopen then null ;
    else open cname ;
    end if ;
    fetch cname into empdetail ;
    if cname % rowcount > 0 then
        while cname % found
            loop
                dbms_output.put_line ( empdetail.ename
                fetch cname into empdetail ;
            end loop ;
        close cname ;
    else
        dbms_output.put_line (' Not found') ;
    end if ;
end ;
```

exception handling precaution.

ADM - Application Development and Maintainance

recruiter - Durga Chilikuri

update →
```
begin
    update ~~employee~~ salary set salary = salary /100 ;
    dbms_output.put_line ( sql % ROWCOUNT) ;
end ;
```

procedure in →

```
declare
cursor cname as
begin
    if num >= 0
        dbms....  ('positive');
    else
        dbms....  ('Negative');
    end if;
end;
```

create or replace procedure spc(num IN number) as

→ procedure successfully created

execute spc (30) → positive

OR exec spc(-3) → Negative

in, out →

```
create or replace procedure ABC (n1 in number,
as n2 in number, n3 out number) as
begin
    n3 := n1 + n2;
end;
```

→ procedure created

variable result number;
exec ABC (2,3, :result);

→ procedure success

print result

→ 5 (BIND VAR)

OR

```
declare
    result number;
begin
    spd (2, 3, result);
    dbms... (result);
end;
```

→ 5 (WITHOUT BIND)

→

```
create procedure spd
as
begin
    dbms_output.put_line ('Hey');
end spd;
```

→ procedure created

execute spd OR begin spd ;
↳ Hey ⟷ end;

→ select procedure_name from user_procedures;

→ select text from user_source where name = 'spd' order by line;

→ create or replace procedure spd
as
result number;
begin
result := 10 * 10;
dbms.... (result);
end spd;

→ create or replace procedure spd
as
num number;
begin
num := 10;
if num > 0 then
dbms... ('positive');
else
dbms... ('Negative');
end if
end spd;

in out → create or replace procedure emp (pemp in employees.employee_id %type, psal in out number)
as
minsal number;
begin
select min(salary) into minsal from employees;
if psal < minsal
then psal := psal * 3;
end if;

```
declare
sal number;
begin
sal := &sal;
emp ( &pemp, sal );
dbms... (sal);
end;
```

enter value for sal:
1000
enter value for pemp:
12
3000

exceptions

NO_DATA_FOUND
TOO_MANY_ROWS
OTHERS

→

```
declare
        emp employee%ROWTYPE;
        result number;
begin
        select * into emp from employees
        where department_id = 10;
        result := emp.salary/0;
exception
        when no_data_found then
                dbms... ('No employee exists');
        when too_many_rows then
                dbms... ('More than 1 employee');
        when others then
                dbms... ('Other exception');
end;
```

HTML 5

MEGHDUT
PAGE NO. DATE / /20

html5test.com w3schools.com

Radio button
Text box
Text area
Button / submit / Reset
Checkbox
Images
Tables
Hyperlinks
<h1> <h2> <h3> <P>
<b>   lists

<h1> → <html>

<head>
<title> Page Title </title>
</head>
<body>
<h1> Heading </h1>
 Paragraph
</body>
</html>

Heading
Paragraph

<h> → heading h1, h2, h3, h4, h5, h6 thats it
 → paragraph
<b> → bold

 → break line need not be closed
<a> → hyperlinks
<img> → image
<i> → italic
<u> → underline
<button> → button
<mark> → highlight
 → list
<hr> → horizontal row
<pre> → preformatted text
<sub> → subscript
<sup> → superscript

<a> →

```
<!DOCTYPE html>
<html>
    <body>
        <a href = "abcd.com"> link </a>
    </body>
</html>
```

→ link

<img> →

```
<!DOCTYPE html>
<html>
    <body>
        <img src = "image.jpg" alt = "no image"
    </body>  width = "104" height = "142">
</html>
```

→ no image

(image tags are not to closed ie. self ending tags

<button> →

```
<!DOCTYPE html>
<html>
    <body>
        <button> submit </button>
    </body>
</html>
```

→ [submit]

 →

```
<!DOCTYPE html>
<html>
    <body>
        <h1> heading </h1>
        <h6> <u> short </u> </h6>
        <a href = "abcd.com"> link1 </a>
        <a href = "efg.com"> link2 </a>
        <button> submit1 </button>
        <button> <br> submit2 </button>
    </body>
</html>
```

→

heading
short
link1 link2 [submit1]
[submit2]

 →

```
<!DOCTYPE html>
<html>
<body>
<h2> Unordered list </h2>
<ul>
<li> Coffee </li>
<li> Tea </li>
</ul>
<h2> Ordered list </h2>
<ol>
<li> Coffee </li>
<li> Tea </li>
</ol>
</body>
</html>
```

```
<dl>
<dt> Coffee </dt>
<dd> - black </dd>
<dt> Milk </dt>
<dd> - cold </dd>
</dl>
```

↳ Coffee
- black
Milk
- cold

↳ Unordered list
- Coffee
- Tea

Ordered list
1. Coffee
2. Tea

`<ul style="list-style-type: disc">`	
~~ ~~	
disc •	`<ol type="1">`
circle ○	
square ■	1 – 1 2 3 4 ...
none	A – A B C D ...
	a – a b c d ...
start="32"	I – I II III IV ...
↳ 32 33 34 ...	i – i ii iii iv ...

style →

```
<!DOCTYPE html>
<html>
<body>
<p style="color: black;"> Blue </p>
</body>
</html>
```

↳ **Black**

lang →

```
<!DOCTYPE html>
<html lang="en-US">
<body>
...
</body>
</html>
```

title → <!DOCTYPE html>
<html>
<body>
 move cursor here
</body>
</html>

↳ move cursor here
I'm a tooltip

<p title = 'I'm a tooltip'> move cursor here

'I'm "a tool"tip'
"I'm 'a too'tip"

I'm "a tool" tip
I'm 'a tool' tip

'I'm "a "to"ol" tip'
"I'm 'a 'to'ol' tip"

I'm "a "to"ol" tip
I'm 'a 'to'ol' tip

similiar quote inside
different quote outside

style → <h1 style = "font-size: 60px;">heading </h1>

<hr> → <h1> line 1 </h1>
<hr>
 Text

↳ Line 1
Text

meta → <head>
<title> First </title>
<meta charset = "UTF-8">
</head>

19/09/2018

 → ignores a lot of space and a number of lines,
 considering it the same.

↳ ignores a lot of space and a number of lines, considering it the same.

<pre> → <pre> ignores not a lot of space and a number
 of lines, considering it different. </pre>

↳ ignores not a lot of space and a number of lines, considering it different.

style → <tagname style = "property : value;" >

property = color : red
font-size : 60%
background-color : blue
font-family : courier
text-align : right
border : 2px

→
```
<!DOCTYPE html>
<html>
<body style = "background-color : lightgreen;" >
<h1 style = "color : blue;" > Heading </h1>
<p style = "font-size : 40px;" > Paragraph </p>
<p style = "font-family : verdana;" > P2 </p>
<p style = "font-size : 160%;" > P3 </p>
<p style = "text-align : center;" > P4 </p>
</body>
</html>
```

→	Bold	**Bold**
	Italic	*Italic*
<u>	<u> Underline </u>	underline
<sub>	_{subscript}	subscript
<sup>	^{superscript}	superscript

```
<strong> Important </strong>      Important
<em> Emphasized </em>            Emphasized
<mark> Marked </mark>            Marked
<small> small </small>           small
<del> Deleted </del>             Deleted
<ins> Inserted </ins>            Inserted
</pre>
```

→ `<p style = " border : 5px solid red; "> Hey </p>`
`<p style = " border : 10 px dotted blue; "> Hi </p>`

→ rgb - red green blue (0, 255, 0)
hsl - hue saturation lightness (360, 100%, 0%)
hex - hexadecimal values (#00ff00)

hsl : h - 0 to 360 0 red 120 green 240 blu
s - 0% to 100% gray to color
l - 0% to 100% 0 ~~black~~ 100 white

rgba - red green blue alpha (opacity)
0 full transparent
1 not transparent

hsla - hue saturation lightness alpha

links →

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      a: link { color : green;
                background - color : red;
                text - decoration : none; }
      a: visited { color : blue;
                   background - color : yellow;
                   text - decoration : bold; }
```

```
a:hover { color: red;
          background-color: green;
          text-decoration: underline; }

a:active { color: yellow;
           background-color: blue;
           text-decoration: italic; }
</style>
</head>
<body>
    <h3> Link Colors </h3>
    <a href = "image.asp" target = "_blank"
        Images </a>
</body>
</html>
```

_blank - opens link in new window
_self - opens link in same window
_parent - opens link in parent frame
_top - opens link in full body of window
framename - opens link in named frame

table → th - table heading
tr - table row
td - table data

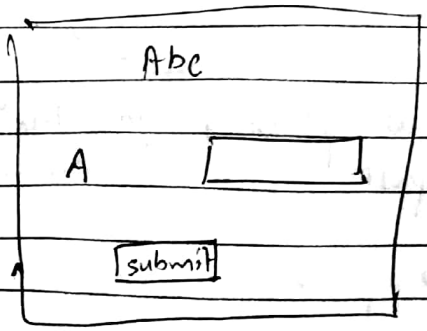
```
<!DOCTYPE html>
<html>
  <body>
    <h3> Tables </h3>
    <table style = "width: 50%">
      <tr>
        <th> Name </th>
        <th> Age </th>
      </tr>
      <tr>
        <td> Raj </td>
```

```
<td> 40 </tr>
</tr>
<tr>
  <td> Ram </td>
  <td> 36 </td>
</tr>
</table>
</body>
</html>
```

Tables

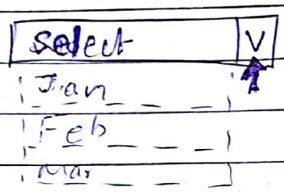
Name	Age
Raj	40
Ram	36

```
<!DOCTYPE html>
<html> <head>
<title> Form </title>
<style> th, td { padding: 15px; } </style>
</head>
<body>
<form action = "https://www.abc.com"
method = "post">
<table align = "center" style = "border:
black double;">
<tr>
<th colspan = "2"> Abc </th>
<tr>
<td> A </td>
<td> <input type = "text" name =
"a" required = "true" /> </th>
<th colspan = "2"> <input type =
"submit" value = "sub" /> </th>
</table>
</form>
</body>
</html>
```



placeholder, size

→ <input list = "months" name = "mon" placeholder = "select"
size = "10" required = "true" >
<datalist id = "months" >
<option value = "Jan" >
<option value = "Feb" >
<option value = "Mar" >
</datalist>

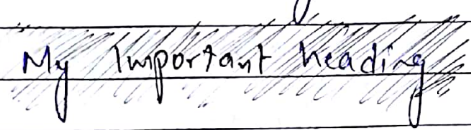


→ <marquee> Please fill </marquee>

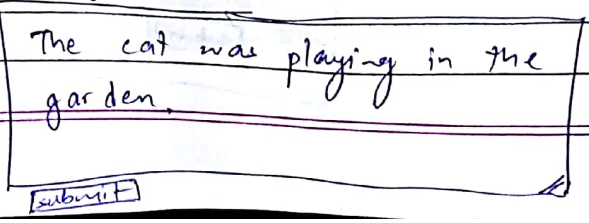
radio → <form>
<input type = "radio" name = "gender" value = "m" />
male
<input type = "radio" name = "gender" value = "f" /> female
<input type = "radio" name = "gender" value = "o" /> other

  → <body>
<div style = "background-color: black; colour: white;
padding: 20px; " >
<h1> My Important
 heading </h1>

</body>



textbox → <textarea name = "msg" rows = "10" cols = "30" > The cat
was playing in the garden. </textarea>
<input type = "submit" >



Java Script

MEGHDUT
PAGE NO. DATE / /20

Obpl - Object Based Programming Language

1. Mobile development
2. web development
3. virus development

jvm - java virtual machine
jsvm - javascript virtual machine

V8 engine is used in google chrome
spider monkey engine is used in mozilla firefox
chakra in ~~microsoft~~ internet explorer.

JRE - java engine
JSE - java script engine } used in browser to run java

html is used to create static page
javascript is dynamic

SDK - software development kits

JS → FJS functional java script
→ OOJS object oriented java script

disadvantage - no security, no io operation
no multithreading, no networking programm

JS framework - 1. Node.js (stable FW)
~~testing~~ 2. Back Bone
3. React
4. React Native
5. Angular * hyderabad
6. Ember * chennai

testing framework - 1. Karma * 4. Mocha
2. Jasmine
3. Protractor * 8.

20.

continuous threading - Jenkins
code quality testing - Sonar qube / PMD
IDE - eclipse / brackets / subline / intelliJ / virtual studio code...

java current version - ECMA8 (released in 2016) but its not stable.

stable one is ECMA6

ECMA - europian computer manufacturing association

alternative of JS - Type script (Micro soft)
Coffee script
Dart
:

best combination is Angular with type Script as all the failures can be overcome.

Programing language - client side (JS)
server side

Browser is also called user agent

JS can be run in any system, no compiler used, cross-platform

RIA - Richer Information Application

→ <SCRIPT>
JS statements... → prefered
</SCRIPT>

OR

<SCRIPT LANGUAGE = "JavaScript 1.2"> → not prefered

inline javascript
external javascript → preferred

javascript 1.2 = ECMA 2 = javascript 2

→ document.write("<...>") (similar to printf in c)
object method

→
```
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript">
      function ms()
      {
        alert("This will be shown")
      }
    </script>
  </head>
  <body onload="ms()">
  </body>
</html>
```
function definition
predefined function for dialogue box or popup
function calling

JS is single threaded model.
ie popup has higher priority than io operations

→
```
<html>
  <body>
    <script>
      document.write("Hey there!")
      alert("Click on ok!")
    </script>
  </body>
</html>
```

```
<html>
  <body>
    <script>
      function abcd()
      { var name;
      name = "srini";
      document.write(name);
      name = "vas";
      document.write("<br/>");
      document.write(name);
      }
    </script>
    <p> Random paragraph </p>
    <button name = "abcd" onclick = "abcd()"> abcd
      calling </button>
  </body>
</html>
```

Random paragraph

abcd calling

srini
vas

Bootstrapping uses onloading [onload = "abcd()"]

usually used in <body onload ... >

avoid using onload

Bootstrapping is if A is activated, B will automatically be activated, then C then D, all on their own.

this is lifecycle method

datatype concept is not present in JS. Everything is a function.

with node.js, even browser is not needed. ie. it is serverless ~~deployment~~ application development

loops
condition
arrays (3 question in exam)

→
```
<html>
  <body>
    <script type = "text/javascript">
    var d = new Date();
    var time = d.getHours();
    if ( time < 10)
    { document.write ("Good Morning "); }
    else
    { document.write ("Good day "); }
    </script>
  </body>
</html>
```

case sensitive
getHours
getDay
getMinutes

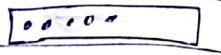
20/09/2018

HTML

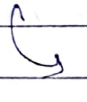
multiple select →
```
<form action = "capgemini.com">
  <select name = "cars" size = "4" multiple>
    <option value = "volvo">Volvo</option>
    <option value = "s">Saab</option>
    <option value = "f">Fiat</option>
  </select>
  <input type = "submit">
</form>
```

range →
```
<input type = "range" id = "a" name = "a" />
```


number →
```
<input type = "number" id = "b" name = "b" />
```

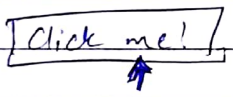

password →
```
<input type = "password" name = "psw" />
```


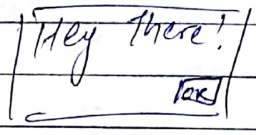
reset →
```
<input type = "reset">
```

checkbox →
```
<form>
  <input type = "checkbox" name = "v1" value = "B" /> Bike
  <input type = "checkbox" name = "v2" value = "C" /> Car
```


☐ Bike
☐ car

onclick →
```
<input type = "button" onclick = "alert('Hey there!')"
  value = "Click me!" />
```





color →
```
<input type = "color" name = "fave" value = "#ff0000" />
```

date →
```
<input type = "date" name = "bday">
```

type = "datetime-local"
= "email"
= "file"
= "month"
= "search"
= "tel"
= "time"
= "url"
= "week"

MEGHDUT

PAGE NO. DATE / /20

attributes →

value = "Abc"
readonly
disabled
size = "40"
maxlength = "10"
min = "30"
max = "50"
autocomplete = "off"
novalidate
autofocus
height = "50"
width = "30"
required = "true"

background image →

```
<p style = "background-image : url('clouds.jpg')">
Surprise </p>
```

JS

PAGE NO. DATE / 120

JS Strings
JS Arrays
JS Arrays Methods
JS Arrays sort
JS Arrays Iteration

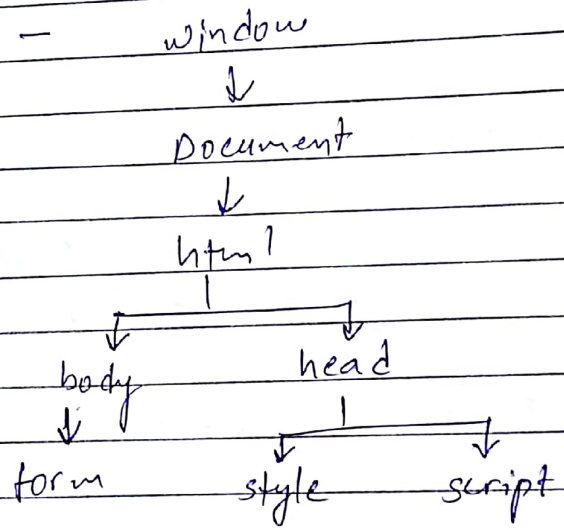
loops
conditions
arrays

DOM - Document Object Modelling

1. html dom
2. xml dom
3. js dom

DOM is used for parsing (splitting) of data.

DOM html -



document.getElementById()
ByName()
ByTgName()

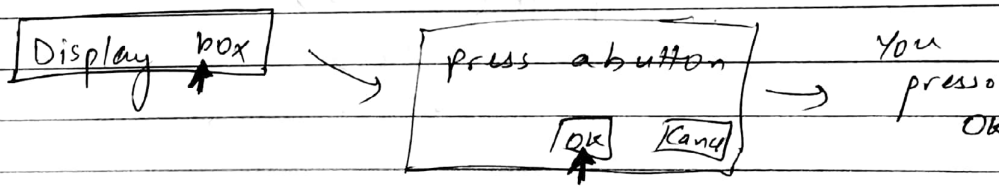
id for CSS
name for JS DOM
tag name for html

length of variable → var_name.length

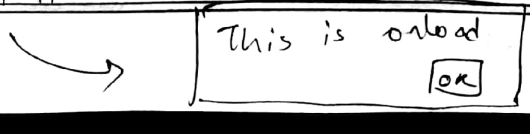
push
pop
delete

display 2 tables using JS.

```
<html>
  <head>
    <script type = "text/javascript">
      function disp()
      {
        var res = confirm("Press a button");
        if (res = true)
        {
          document.write("You pressed OK");
        }
        else
        {
          document.write("You pressed cancel")
        }
      }
    </script>
  </head>
  <body>
    <input type = "button" onclick = "disp()"
      value = "Display box" />
  </body>
</html>
```




```
<html> <head>
  <script type = "text/javascript">
    function msg()
    { alert("This is onload"); }
  </script> </head>
  <body onload = "msg()"> </body>
</html>
```



→
```
<html>
  <head>
    <script src = " comm. js " >
    </script>
  <body>
    <script>
      document. write ( " The msg is " + msg + " BYE ");
    </script>
  </body>
</html>
```

↳ The msg is Hey BYE

→
```
<html>
  <body>
    <script>
      document. write ( " Continue " );
      var i = 0;
      for ( i = 0; i <= 5; i++ )
      { if ( i == 3 )
          continue;
        document. write ( " val = " + i + " <br> " );
      }
      document. write ( " <br><br> " );
      for ( i = 0; i < 5; i++ )
      { if ( i == 3 )
          break;
        document. write ( " val = " + i + " <br> " );
      }
    </script>
    <!-- comments --> <p> Outside script </p>
  </body>
</html>
```

↳

val = 0	val = 0
val = 1	val = 1
val = 2	val = 2
val = 4	outside script

```
<html>
  <body>
    <script>
      function prod(a,b)
      { return a*b; }
      document.write(prod(4,3));
    </script>
  </body>
</html>
```

→ 12