**TERADATA ASSIGNMENTS**

**LAB 1**

**AGGREGATION FUNCTION**

select avg(salary) as "AVG"

,count(salary) as salcnt

,count(\*) as rowcnt from AGGREGATION\_125000;

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

select min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table;

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

select dept\_no

,min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table;

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

select dept\_no

,min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table group by(dept\_no);

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

select deptno

,min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table group by 1

order by 1;

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

select deptno

,min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table where deptno in (200,400)

group by deptno

order by 1;

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

select deptno

,min(salary)

,max(salary)

,avg(salary)

,count(\*)

from employee\_table where deptno in (200,400)

group by deptno

having count(\*)>1;

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

select product\_id

,extract (month from sale\_date) as mth

,extract (year from sale\_date) as yr

,sum(daily\_sales) as sum\_daily\_sales

from sales\_table

group by grouping sets(product\_id,mth,yr)

order by product\_id desc,mth desc,yr desc;

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

select product\_id

,extract (month from sale\_date) as mth

,extract (year from sale\_date) as yr

,sum(daily\_sales) as sum\_daily\_sales

from sales\_table

group by rollup(product\_id,mth,yr)

order by product\_id desc,mth desc,yr desc;

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

select product\_id

,extract (month from sale\_date) as mth

,extract (year from sale\_date) as yr

,sum(daily\_sales) as sum\_daily\_sales

from sales\_table

group by cube(product\_id,mth,yr)

order by product\_id desc,mth desc,yr desc;

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

select dept\_no, avg(salary), sum(salary)

from employee\_table

where dept\_no in (200,300,400)

group by dept\_no

having Avg(salary)>43000

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

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

**BASIC SQL FUNCTIONS**

select first\_name

,last\_name

,class\_code

,grade\_pt

from student\_table;

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

select \* from student\_table

order by last\_name;

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

select \* from student\_table

order by 5;

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

select \* from student\_table

order by grade\_pt;

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

select \* from student\_table

order by last\_name desc;

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

select \* from student\_table

order by 5 desc;

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

select \* from student\_table

order by grade\_pt desc;

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

select \* from student\_table

order by class\_code desc,

grade\_pt asc;

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

select employee\_no

,dept\_no

,first\_name

,last\_name

,salary

from employee\_table

order by dept\_no desc

,last\_name asc

,salary desc;

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

select \* from student\_table

order by class\_code;

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

select \* from student\_table

order by case class\_code

when 'fr' then 1

when 'so' then 2

when 'tr' then 3

when 'sr' then 4

else 5

end;

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

select first\_name as fname

,last\_name lname

,class\_code "class code"

,grade\_pt as "avg"

,student\_id as stud\_id

from student\_table;

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

select first\_name,last\_name,class\_code,grade\_Pt

from student\_table;

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

select 'character data'

,'character dat' (tittle 'characyter//data')

,123 (tittle 'numeric data')

;'character data'(tittle 'my//stacked//example');

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

select student\_id

,last\_name

,first\_name

,class\_code as sum

,grade\_pt

from student\_table

where grade\_pt>3.6

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

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

**COLLECTION STATISTICS**

expalin select \* from new\_employee\_table;

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

collect statistics on new\_employee\_table

column employee\_no;

explain select \* from new\_employee\_table;

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

collect statistics on new\_employee\_table

column dept\_no;

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

collect statistics on new\_employee\_table

index(first\_name,last\_name);

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

collect statistics on new\_employee\_table

column (employee\_no,dept\_no);

collect statistics on employee\_table;

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

collect statistics on new\_employee\_table

column(first\_name, last\_name) on

employee\_table;

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

collect statistics on new\_employee\_table

column (first\_name) on

employee\_table;

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

collect statistics column

(dept\_no)

on employee\_table;

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

select \* from

sales\_data\_mart

where sale\_date between

'2013-01-01' and '2013-01-01';

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

select \* from sales\_table

where product\_id = 1000;

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

select \* from sales\_table

where product\_id = 1000

and daily\_sales=50000;

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

create table employee\_table\_new as employee\_table

withdata

and statistics;

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

collect statistics using system sample column (product\_id) on sales\_table;

collect statistics using system sample 15 percent and maxintervals 10, column (product\_id) as product\_stats on sales\_table;

collect statistics using system sample 20 percent and maxintervals 25, column (product\_id) as product\_stats on sales\_table;

show statistics on sales\_table;

show statistics values column product\_id on sales\_table;

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

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

**DATA MANIPULATION LANGUAGE**

insert into my\_table(column2,column1,column3,column6)

values(124.56,'my character', 12587,'2000-12-31');

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

insert into my\_table

select \* from my\_original\_table;

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

insert into my\_table

select \* from my\_original\_table1;

insert into my\_table

select \* from my\_original\_table2;

insert into my\_table

select \* from my\_original\_table3;

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

insert into my\_table

select \* from my\_original\_table1;

union

select \* from my\_original\_table2;

union

select \* from my\_original\_table3;

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

update my\_table

set

column2 = 256

,column4 = 'mine'

,column5 = 'yours'

where column1 = 'my character data';

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

update my\_table

set

column2 = column2 + 256

where column1 = 'my character data'

and column4 = 'mine'

and column5 = 'yours';

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

update my\_table

'set column 3 = 20000000

where column2 in (select column2 from ct1\_tb1

where column3 > 5000

and et1\_tb1.column4 is not null);

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

**LAB2**

**DISTINCT VS GROUP BY**

select distinct class\_code from student\_table

order by 1;

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

select distinct class\_code from student\_table

group by 1

order by 1;

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

select distinct class\_code, grade\_pt

from student\_table

order by class\_code,grade\_pt;

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

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

**EXPLAIN**

explain select \* from employee\_table;

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

explain select \* from employee\_table

where employee\_no=2000000;

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

explain select \* from sales\_table

where product\_id =1000;

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

create unique index(first\_name,last\_name) on employee\_table;

expalin select \* from employee\_table

where first\_name='loraine'

and last\_name='larkins';

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

explain select E.\*,D.\*

from employee\_table as E inner join

department\_table as D on E.Dept\_no=D.Dept\_no;

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

explain select E2.\*,D.\*

from employee\_table2 as E2 inner join

department\_table as D on E2.Dept\_no=D.Dept\_no;

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

explain select E.\*, D.\*

from employee\_table as E, Department\_table2 as D

where E.Dept\_no=D.Dept\_no;

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

explain select \* from addresses;

collect statistics on addresess

column subscribe\_no;

explain select \* from addresess;

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

expalin select \* from student\_table s,

course\_table c,student\_course\_table sc

where s.student\_id=sc.student\_id;

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

explain select \* from employee\_table as e

where salary=36000

and dept\_no=400;

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

explain select \* from order\_ppi

where order\_date='1998-05-04';

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

explain select \* from order\_ppi

where order\_date BETWEEN '1998-05-01' and '1998-05-31';

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

explain select \* from order\_ppt

where order\_date=current\_date;

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

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

**FORMAT FUNCTIONS**

select current\_date(format 'mm-dd-yy');

select 5133000346(format '99-999-9999');

select 5133000346(format '99-999-9999');

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

select 'theymatch' as "do they"

where 'A'(casespecific)='a'(cs);

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

select 'theymatch' as "Do they"

where 'A'='a';

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

select LOWER('ABCDE') as result1;

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

select 'theymatch' as "do they"

where lower('abcde')='abcde';

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

select UPPER('ABCDE') as result1;

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

select 'they match' as "do they"

where 'ABCDE'=UPPER('abcde');

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

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

**HELP AND SHOW**

select \* from dbc.dbcinfo;

help table sql\_class.employee\_table;

slect session;

select account

,database

,session

,user

help session;

help'sql';

help 'sql csum';

show table sql\_class.employee\_table;

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

select set table sql\_class.employee\_table,no fallback,

no before journal,

no after journal,

checksum = default

(

employee\_no integer,

dept\_no smallint,

last\_name char(20) character set latin not casespecific,

first\_name varchar(12) character set latin not casespecific,

salary decimal(8,2)

unique primary index(employee\_no)

index(last\_name)

index(dept\_no);

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

show view sql\_views.employee\_v;

show macro my\_mac;

create macro mjl01.my\_mac (inparm1 integer, inparam2 char(10))

as (select dept,day\_of\_week,avg(sal)

from sys\_calendar.calendar sc, mytable

where calender\_date = :inparm2 (date, format 'yyyymmdd')

and dept=:inparm1

groupby 1,2);

show trigger avg\_sal\_t;

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

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

**JOIN FUNCTIONS**

select cutomer\_table.customer\_number,customer\_name,order\_number.order\_total

from customer\_table,order\_table

where customer\_table.customer\_number=order\_table.customer\_number;

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

select cutomer\_table.customer\_number,customer\_name,order\_number.order\_total

from customer\_table as cust,order\_table as ord

where cust.customer\_number=ord.customer\_number;

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

select cutomer\_table.customer\_number,customer\_name,order\_number.order\_total

from customer\_table as cust inner join order\_table as ord

on cust.customer\_number=ord.customer\_number;

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

select first\_name, last\_name,

E.dept\_no department\_name from

employee\_table as E

innerjoin department\_table as D

on E.dept\_no=D.dept\_no;

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

select first\_name, department\_name

from employee\_table as E

right outer join department\_table as D

on E.dept\_no=D.dept\_no;

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

select first\_name, department\_name

,department\_name

from employee\_table as E

full outer join department\_table as D

on E.dept\_no=D.dept\_no;

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

select first\_name, department\_name

from employee\_table as E,

department\_table as D

where E.dept\_no=D.dept\_no

AND Department\_Name like 'marke%';

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

select first\_name, department\_name

from employee\_table as E

left outer join department\_table as D

on E.dept\_no=D.dept\_no

and E.dept\_no=100;

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

select first\_name, department\_name

from employee\_table as E,

left outer join department\_table as D

on E.dept\_no=D.dept\_no

where Department\_Name like 'marke%'

order by 1,2,3;

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

select first\_name, department\_name

from employee\_table as E,

department\_table as D;

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

select first\_name, department\_name

from employee\_table as E,

inner join department\_table as D;

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

select customer\_name,

order\_number

from customer\_table

cross join

order\_table

where

order\_number=123456

order by 1;

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

select mgrs.deptno.mgrs.last\_name as mgrname.mgrs.salary as mgrsal.

emps.last\_name as empname,emps.salary as empsal

from employee\_table2 as emps

inner join

employee\_table2 as mgrs

on emps.dept\_no=mgrs.dept\_no

where mgrs.mgr='y'

and emps.salary>mgrs.salary;

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

select customer\_name,

order\_number,

order\_total

from customer\_table as cust

innerjoin order\_table as ord

on cust.customer\_number

=ord.customer\_number

where

customer\_name like 'billy%' order by 1;

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

select customer\_name,

order\_number,

order\_total

from customer\_table as cust

innerjoin order\_table as ord

on cust.customer\_number

=ord.customer\_number

and

customer\_name like 'billy%' order by 1;

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

select customer\_name,

order\_number,

order\_total

from customer\_table as cust

left outer join order\_table as ord

on cust.customer\_number

=ord.customer\_number

where

customer\_name like 'billy%' order by 1;

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

select customer\_name,

order\_number,

order\_total

from customer\_table as cust

left outer join order\_table as ord

on cust.customer\_number

=ord.customer\_number

and

customer\_name like 'billy%' order by 1;

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

traditional syntax:

select s.\*,c.\*

from student\_table as s,

course\_table as c,

student\_course\_table as sc

where s.student\_id=sc.student\_id

and c.course\_id=sc.course\_id;

ANSCI syntax

select s.\*,c.\*

from student\_table as s,

inner join

student\_course\_table as sc

s.student\_id=sc.student\_id

on

inner join

course\_table as c

on c.course\_id=sc.course\_id;

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

select s.\*,c.\*

from student\_table as s,

inner join

student\_course\_table as sc

inner join

course\_table as c

on c.course\_id=sc.course\_id

on s.student\_id=sc.student\_id;

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

select

cla1.\*, sub1.\*, add1.\*, pro1.\*, ser1.\*

from claims as clal

inner join

subscribers as sub1

cla1.subscriber\_no=sub1.subscriber\_no

cla1.member\_no=sub1.member\_no

and

inner join

addresess as add1

sub1.subscriber\_no = add1.subscriber\_no

on inner join

providers as pro1

cla1.provider\_no = pro1.provider\_code

on inner join

service as ser1

on cla1.claim\_service = ser1.service\_code;

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

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

**JOIN INDEX**

select join index emp\_dept\_idx as select employee\_no

,e.dept\_no

,first\_name

,last\_name

,salary

,department\_name

from employee\_table a E

inner join

department\_table as D

on E.dept\_no=D.dept\_no

primary index(employee\_no);

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

select join index emp\_dept\_idx as select employee\_no

,e.dept\_no

,first\_name

,last\_name

,salary

,department\_name

from employee\_table a E

left outer join

department\_table as D

on E.dept\_no=D.dept\_no

primary index(employee\_no);

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

create join index cust\_order\_idx as

select

(c.customer\_number,customer\_name),

(order\_number,order\_date,order\_total)

from customer\_table as c

inner join

order\_table as o

on c.customer\_number=

o.customer\_number

primary index(customer\_number);

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

select join index employee\_idx as select employee\_no

,dept\_no

,first\_name

,last\_name

,salary

from employee\_table

primary index(dept\_no);

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

create join index agg\_order\_idx as select

customer\_number

,extract(year from order\_date) as yr

,extract(month from order\_date) as mon

,count(\*) as country

,sum(order\_total) as sunny

from order\_table

group by 1,2,3;

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

create join index agg\_order\_idx as select

customer\_number

,extract(year from order\_date) as yr

,extract(month from order\_date) as mon

,count(\*) as country

,sum(order\_total) as sunny

from order\_table

group by 1,2,3;

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

create join index agg\_order\_idx as select

customer\_number

,extract(year from order\_date) as yr

,extract(month from order\_date) as mon

,count(\*) as country

,sum(order\_total) as sunny

,max(order\_total) as max\_ord

,min(Order\_total) as min\_ord

from order\_table

group by 1,2,3;

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

select join index emp\_dept\_glob as select employee\_no

,e.dept\_no

,first\_name

,last\_name

,e.rowid as empri

,department\_name

,d.rowid as deptri

from employee\_table a E

inner join

department\_table as D

on E.dept\_no=D.dept\_no

primary index(dept\_no);

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

create hash index emp\_hash\_idx

(dept\_no,first\_name,last\_name)

on employee\_table;

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

create hash index emp\_hash\_idx

(dept\_no,first\_name,last\_name)

on employee\_table

order by value (dept\_no);

**LAB 3**

SELECT (2+4)\*5

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

SELECT 2+4/5

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

SELECT 2+4.0/5

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Sale\_Date) AS "CSum"

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Sale\_Date) AS "CSUM"

FROM Sales\_Table WHERE Product\_ID BETWEEN 1000 and 2000;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Product\_ID, Sale\_Date)

AS "CSum" FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Sale\_Date) AS "CSUM"

FROM Sales\_Table WHERE Product\_ID BETWEEN 1000 and 2000 ORDER BY Daily\_Sales;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Product\_ID, Sale\_Date)

AS "CSum" FROM Sales\_Table GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Product\_ID, Sale\_Date)

AS "CSum", CSUM(1, Product\_ID, Sale\_Date) as "Seq\_Number" FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, CSUM(Daily\_Sales, Product\_ID, Sale\_Date)

AS "CSum", CSUM(1, Product\_ID, Sale\_Date) as "Seq\_Number"

FROM Sales\_Table GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales, SUM(Daily\_Sales) OVER(ORDER BY Sale\_Date

ROWS UNBOUNDED PRECEDING) AS SUMOVER

FROM Sales\_Table WHERE Product\_ID BETWEEN 1000 and 2000;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS SUMOVER

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS SUMOVER

SUM(1) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS Seq\_Number

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS SUMOVER

FROM Sales\_Table GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS SumANSI

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS Subtotal

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS GRANDTotal

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MSUM(Daily\_Sales, 3, Product\_ID, Sale\_Date)as Msum3\_Rows

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MSUM(Daily\_Sales, 3, Product\_ID, Sale\_Date)as MSum3\_Rows

FROM Sales\_Table GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MSUM(Daily\_Sales, 3, Product\_ID, Sale\_Date)as MSum3

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MSUM(Daily\_Sales, 3, Product\_ID, Sale\_Date) AS MSum3,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID,

Sale\_Date ROWS 2 Preceding) AS SUM3\_ANSI

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS 2 Preceding) AS SUM3,

SUM(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED Preceding) AS Continuous

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

SUM(Daily\_Sales) OVER(ORDER BY Product\_ID, Sale\_Date

ROWS 2 Preceding) AS SUM3,

SUM(Daily\_Sales) OVER (PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED Preceding) AS Continuous

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MAVG(Daily\_Sales, 3, Product\_ID, Sale\_Date)AS AVG3\_Rows

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MAVG(Daily\_Sales, 3, Product\_ID, Sale\_Date)AS AVG3\_Rows

FROM Sales\_Table GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MAVG(Daily\_Sales, 3, Product\_ID, Sale\_Date)AS AVG\_3,

AVG(Daily\_Sales) OVER (ORDER BY Product\_ID,

Sale\_Date ROWS 2 Preceding) AS AVG\_3\_ANSI

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

AVG(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS 2 Preceding) AS AVG\_3\_ANSI

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

AVG(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS 2 Preceding) AS AVG3,

AVG(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED Preceding) AS Continuous

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

AVG(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS 2 Preceding) AS AVG3,

AVG(Daily\_Sales) OVER (PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED Preceding) AS Continuous

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MDIFF(Daily\_Sales, 4, Product\_ID, Sale\_Date) as "MDiff"

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MDIFF(Daily\_Sales, 7, Product\_ID, Sale\_Date) as Compare2Rows

FROM Sales\_Table

GROUP BY Product\_ID;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

ROW\_NUMBER() OVER (PARTITION BY Product\_ID ORDER BY Sale\_Date

RESET WHEN Daily\_Sales<= SUM(Daily\_Sales)

OVER (PARTITION BY Product\_ID ORDER BY Sale\_Date

ROWS BETWEEN 1 PRECEDING AND 1 PRECEDING)) -1 as Increases

FROM Sales\_Table WHERE Product\_ID Between 1000 and 2000;

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

SELECT Product\_ID, Month(Sale\_Date) as Mo, sum(Daily\_Sales) as Monthly\_Sum,

ROW\_NUMBER() OVER (PARTITION BY Product\_ID ORDER BY Mo

RESET WHEN sum(Daily\_Sales)<=

SUM(sum(Daily\_Sales)) OVER (PARTITION BY Product\_ID ORDER BY Mo

ROWS BETWEEN 1 PRECEDING AND 1 PRECEDING)) -1 as Balance\_Increase

FROM Sales\_Table GROUP BY Product\_ID, Mo;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales) AS "RANK"

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS "RANK"

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS Rank1,

RANK(-Daily\_Sales) AS Rank2

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK() OVER(ORDER BY Daily\_Sales) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK() OVER(ORDER BY Daily\_Sales DESC) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank1 < 7;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK() OVER (PARTITION BY Product\_ID ORDER BY Daily\_Sales DESC) AS Rank1

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank1 < 4;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS Rank1,

RANK(-Daily\_Sales) AS Rank2

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank(-Daily\_Sales) < 6;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS Rank1,

RANK(-Daily\_Sales) AS Rank2

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank2 < 6;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank1 < 6;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank1 < 6;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales ASC) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank1 < 2;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

QUALIFY Rank(Daily\_Sales ASC) < 6;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

RANK(Daily\_Sales) AS Rank1

FROM Sales\_Table

WHERE Product\_ID IN (1000, 2000)

GROUP BY Product\_ID

QUALIFY Rank1 < 4;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

PERCENT\_RANK() OVER (PARTITION BY PRODUCT\_ID

ORDER BY Daily\_Sales DESC) AS PercentRank1

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

PERCENT\_RANK() OVER (ORDER BY Daily\_Sales DESC) AS PercentRank1

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

PERCENT\_RANK() OVER (ORDER BY Daily\_Sales DESC) AS PercentRank1

FROM Sales\_Table;

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

COUNT(\*) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS Seq\_Number

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

COUNT(\*) OVER (ORDER BY Product\_ID, Sale\_Date) AS No\_Seq

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

COUNT(\*) OVER (PARTITION BY PRODUCT\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS StartOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MAX(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS MaxOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MAX(Daily\_Sales) OVER (PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS MaxOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MIN(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS MinOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MIN(Daily\_Sales) OVER (ORDER BY Product\_ID, Sale\_Date) AS MinOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

MIN(Daily\_Sales) OVER (PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date

ROWS UNBOUNDED PRECEDING) AS MinOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

ROW\_NUMBER() OVER (ORDER BY Product\_ID, Sale\_Date) AS Seq\_Number

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT Product\_ID, Sale\_Date, Daily\_Sales,

ROW\_NUMBER() OVER (PARTITION BY Product\_ID

ORDER BY Product\_ID, Sale\_Date) AS StartOver

FROM Sales\_Table WHERE Product\_ID IN (1000, 2000);

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

SELECT First\_Name, CHARACTERS(First\_Name) AS Lnth

FROM Employee\_Table WHERE CHARACTERS(First\_Name)< 7

ORDER BY 1;

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

SELECT Last\_Name, CHARACTERS(Last\_Name) AS Lnth

FROM Employee\_Table ORDER BY 1;

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

SELECT Last\_Name, CHARACTERS(TRIM(Last\_Name)) AS C\_Length

FROM Employee\_Table ORDER BY 1;

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

SELECT First\_Name,

Trim(trailing 'Y' from First\_Name) AS No\_Y

FROM Employee\_Table

ORDER BY 1;

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

SELECT First\_Name,

SUBSTRING(First\_Name FROM 2) AS GoToEnd

FROM Employee\_Table;

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

SELECT First\_Name,

SUBSTRING(First\_Name FROM 0 For 6) AS Before1

FROM Employee\_Table;

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

SELECT First\_Name,

SUBSTRING(First\_Name FROM -1 For 3) AS Before2

FROM Employee\_Table;

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

SELECT First\_Name,

SUBSTRING(First\_Name FROM 3 For 0) AS WhatsUp

FROM Employee\_Table;

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

SELECT Dept\_No,

Department\_Name as Depty,

SUBSTRING(Depty FROM 1 FOR POSITION(' ' IN Department\_Name) -1) as Word1

FROM Department\_Table;

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

SELECT DISTINCT Department\_Name as Dept\_Name,

SUBSTRING(Department\_Name FROM

POSITION(' ' IN Department\_Name) +1)as Word2

FROM Department\_Table

WHERE POSITION(' ' IN trim(Department\_Name)) >0;

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

CREATE MULTISET TABLE Property\_Owners

(

Cust\_No INTEGER

,Prop\_No INTEGER

,Prop\_Val\_Time PERIOD (DATE) NOT NULL as VALIDTIME

,Prop\_Tran\_Time PERIOD (TIMESTAMP(6) with TIME ZONE)

NOT NULL as TRANSACTIONTIME

) PRIMARY INDEX(Prop\_No);

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

INSERT INTO PROPERTY\_OWNERS

(Cust\_No, Prop\_No)

VALUES (1, 100);

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

UPDATE Property\_Owners

SET Cust\_No= 2

WHERE Prop\_No = 100;

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

NONSEQUENCED VALIDTIME

SELECT \* FROM Property\_Owners;

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

VALIDTIME AS OF DATE '2011-01-30'

SELECT \* FROM Property\_Owners;

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

SELECT \* FROM (SELECT AVG(salary) FROM Employee\_table) TeraTom(avgsal);

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

SELECT Last\_name, Salary, Avgsal

FROM (SELECT AVG(salary) FROM Employee\_table) AS TeraTom(avgsal)

INNER JOIN Employee\_table ON avgsal < salary;

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

SELECT Last\_name, Salary, Avgsal, Maxsal

FROM (SELECT AVG(salary) as avgsal, Max(Salary) as maxsal

FROM Employee\_table) AS TeraTom

INNER JOIN Employee\_table ON avgsal < salary;

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

SELECT Last\_name, Dept\_No, Salary, Avgsal

FROM (SELECT Dept\_No as Depty, AVG(salary) as avgsal

FROM Employee\_table GROUP BY 1) AS TeraTom

INNER JOIN Employee\_table ON Dept\_No = Depty

WHERE Salary > Avgsal;

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

SELECT Product\_ID AS Product, Cal\_yr,

Sep\_sales AS September\_sales,

Oct\_sales AS October\_sales,

Nov\_sales AS November\_sales

FROM(SELECT Product\_ID, Extract(YEAR FROM Sale\_date) AS Cal\_Yr,

SUM(CASE ((Sale\_date/100) MOD 100)

WHEN 9 THEN Daily\_Sales

ELSE 0

END) AS Sep\_sales,

SUM(CASE((Sale\_date/100) MOD 100)

WHEN 10 THEN Daily\_Sales

ELSE 0

END) AS Oct\_sales,

SUM(CASE((Sale\_date/100) MOD 100)

WHEN 11 THEN Daily\_Sales

ELSE 0

END) AS Nov\_sales

FROM Sales\_table

WHERE Sale\_date BETWEEN 1000901 AND 1001130

GROUP BY 1,2)

DT\_Month\_Sum\_Sales;

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

SELECT Employee\_No, Dept\_No, First\_Name, Last\_Name, Salary, Position\_Name,

Mgr\_Employee\_No, DEPTH+1

from Hierarchy\_DT

(INNER JOIN

Hierarchy\_Table

on Emp=Mgr\_Employee\_No)

SELECT Emp, Dept, FirstN, LastN, Sal, Pos\_Name, Mgr, DEPTH

FROM Hierarchy\_DT

order by DEPTH, Mgr;

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

CREATE VOLATILE TABLE Dept\_Aggreg\_vt, NO LOG

(Dept\_no Integer,

Sum\_Salary Decimal(10, 2),

Avg\_Salary Decimal(7, 2),

Max\_Salary Decimal(7, 2),

Min\_Salary Decimal(7, 2),

Cnt\_Salary Integer)

ON COMMIT PRESERVE ROWS;

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

INSERT INTO Dept\_Aggreg\_vt

SELECT Dept\_no'

SUM(Salary),

AVG(Salary),

MAX(Salary),

MIN(Salary),

COUNT(Salary)

FROM Employee\_Table

GROUP BY Dept\_no;

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

SELECT Department\_Name,

Avg\_Salary,

Max\_Salary,

Min\_Salary,

FROM Dept\_Aggreg\_vt AS VT INNER JOIN Department\_Table D

ON VT.dept\_no = D.dept\_no

WHERE Cnt\_Salary> 1;

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

CREATE GLOBAL TEMPORARY TABLE Dept\_Aggreg\_gt

(Dept\_no Integer,

Sum\_Salary Decimal(10, 2),

Avg\_Salary Decimal(7, 2),

Max\_Salary Decimal(7, 2),

Min\_Salary Decimal(7, 2),

Cnt\_Salary Integer)

ON COMMIT PRESERVE ROWS;

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

INSERT INTO Dept\_Aggreg\_gt

SELECT Dept\_no'

SUM(Salary),

AVG(Salary),

MAX(Salary),

MIN(Salary),

COUNT(Salary)

FROM Employee\_Table

GROUP BY Dept\_no;

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

SELECT \* FROM Dept\_Aggreg\_gt

ORDER BY 1;

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

SELECT Department\_Name,

Avg\_Salary,

Max\_Salary,

Min\_Salary,

FROM Dept\_Aggreg\_gt AS GT INNER JOIN Department\_Table D

ON GT.dept\_no = D.dept\_no

WHERE Cnt\_Salary> 1;

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

**LAB 4**

**WHERE CLAUSE**

select product\_id,sale\_date,daily\_sales

,quantile (100, daily\_sales, sale\_date desc) as " quantile"

from sales\_table

where product\_id < 3000

and sale\_date >= 1000390;

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

select product\_id,sale\_date,daily\_sales

,quantile (100, daily\_sales, sale\_date) as " quantile"

from sales\_table

where product\_id < 3000

and sale\_date > 1000390;

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

select product\_id,sale\_date,daily\_sales

,quantile (100, daily\_sales, sale\_date ) as " percentile"

from sales\_table

qualify "percentile" >=60;

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

select product\_id,sale\_date,daily\_sales

, quantile (100, daily\_sales asc, sale\_date ) as " percentile"

from sales\_table

qualify "percentile" >= 70;

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

select product\_id,sale\_date,daily\_sales

, quantile (100, daily\_sales, sale\_date ) as " percentile"

from sales\_table

qualify "percentile" >= 70

order by "percentile" desc;

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

select product\_id,sale\_date,daily\_sales,

quantile(4,daily\_sales , sale\_date) as "quartiles"

from sales\_table

where product\_id in (1000,2000);

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

select product\_id,sale\_date,daily\_sales,

quantile(4,daily\_sales , sale\_date) as "quartiles"

from sales\_table

where product\_id = 1000;

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

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

**TOP SQL COMMANDS CHEAT SHEET**

SELECT \* FROM EMPLOYEE\_TABLE

ORDER BY DEPT\_NO;

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

SELECT FIRST\_NAME AS FNAME,

LAST\_NAME AS LNAME

,CLASS\_CODE AS "CLASS CODE"

,GRADE\_PT AS "AVG"

,STUDENT\_ID

FROM STUDENT\_TABLE

WHERE FNAME='HENRY';

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

SELECT TOP 7 LAST\_NAME,

CLASS\_CODE,

GRADE\_PT

FROM STUDENT\_TABLE

ORDER BY GRADE\_PT DESC;

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

SELECT \* FROM EMPLOYEE\_TABLE

SAMPLE 8;

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

SELECT \* FROM EMPLOYEE\_TABLE

SAMPLE .50;

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

SHOW TABLE EMPLOYEE\_TABLE;

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

SELECT DEPTNO

,MIN(SALARY) AS "MIN"

,MAX(SALARY) AS"MAX"

,SUM(SALARY) AS "SUM"

,AVG(SALARY) AS "AVG"

,COUNT(\*) AS "CNT"

FROM EMPLOYEE\_TABLE

GROUP BY DEPTNO

ORDER BY 1;

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

SELECT C.CUSTOMER\_NUMBER,C.CUSTOMER\_NAME,

O.ORDER\_NUMBER,O.ORDER\_TOTAL

FROM CUSTOMER\_TABLE AS C

,ORDER\_TABLE AS O

WHERE C.CUSTOMER\_NUMBER=O.CUSTOMER\_NUMBER;

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

SELECT C.CUSTOMER\_NUMBER,C.CUSTOMER\_NAME,

O.ORDER\_NUMBER,O.ORDER\_TOTAL

FROM CUSTOMER\_TABLE AS C

INNER JOIN

ORDER\_TABLE AS O

WHERE C.CUSTOMER\_NUMBER= O.CUSTOMER\_NUMBER;

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

select order\_date

,order\_date(format 'eeeeeebbmmmbbdbyyyy')

,order\_date+60 as"due date"

,order\_total(format '$$$$.$$$$,99')

,add\_months(order\_date,4) as over\_due

,extract(month from order\_date) as themonth

from order\_table;

order by 1;

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

select calender\_date,day\_of\_week,day\_of\_,month

,day\_of\_year,day\_of\_calender,weekday\_of\_month

,week\_of\_month,week\_of\_year,week\_of\_calender

,month\_of\_quater,month\_of\_year,month\_of\_calender

,quarter\_of\_year,quater\_of\_calender,year\_of\_calender

from sys\_calender.calender

where calender\_date='1959-01-01';

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

select o.\*

from order\_tables a o

inner join

sys\_calendar.calendar

on order\_date=calender\_date

and quarter\_of\_year=4

and day\_of\_week=6

and week\_of\_month=0;

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

select order\_total

,order\_total(format '$ZZZ,ZZZ,99') as dollar

,order\_date

,order\_date(format 'mm-dd-yy')

,order\_date(format 'eeebbmmbdd,yyyy') as goodform

from order\_table;

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

select product\_id,sales\_date

,daily\_sales

rank() over(order by daily\_sales desc)

as rank1

from sales\_table;

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

select e.\*,avgsal

from employee\_table as E

inner join

(select dept\_no as dept\_no,avg(salary) as avgsal

from employee\_table

group by dept\_no) as tera tom

on E.dept=teratom.dept\_no

order by E.dept\_no;

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

select \* from customer\_table

where customer\_number in(

select customer\_number

from order\_table);

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

select \* from employee\_table as EE

where salary>(

select avg(salary)

from employee\_table as EEE

WHERE EE.dept\_no=EEEE.DEPT\_NO);

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

select first\_name

substring(first\_name from 2 for 3) as quiz

from employee\_table;

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

create volatile table order\_vol

as(select \* from order\_table

where extract(month from order\_date)=9)

with data and statistics

primary index(customer\_number)

on commit preserve rows;

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

create volatile table dept\_agg\_vol,no log

(dept\_no integer

,sum\_salary decimal(10,2))

on commit delete rows;

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

begi transac tion;

insert into dept\_agg\_vol

select dept\_no,sum(salary) from employee\_table

group by 1;

select \* from dept\_agg\_vol;

et;

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

select counter as "typical rows per produc\_id"

from

(select product\_id,count(\*)

from sales\_table group by 1)

as teratom(col1,counter),

(select count(distinct(product\_id))

from sales\_table) as derived2.num\_row/2;

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

locking row for access

select vproc

,vproc,accountname

,(maxperm)

,(maxspool)

,(maxtemp)

from dbc.diskspace

where databasename='sq01';

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

locking row for access

select

sum(maxperm)

,sum(maxspool)

,sum(maxtemp)

from dbc.diskspace

where databasename=user;

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

select

vproc

,cast(tablename as char(20))

,currentperm

,pealperm

from dbc.tablesizeV

where databasename='sql\_class'

order by tablename,vproc;

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

locking row for access

hashamp(hashbucket(hashrow(employee\_no)))

,count(\*)

from sql\_class.employee\_table

group by 1

order by 1;

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

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

**VIEW FUNCTIONS**

create view department\_salaries as

select dept\_no

,sum(salary) as sumsal

,sum(sal) / 12 as monthsal

from employee\_table

group by 1;

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

replace view department\_salaries as

select dept\_no

,sum(salary) as sumsal

,sum(sal) / 12 as monthsal

from employee\_table;

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

show view employee\_table;

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

create view emp4v4 as

select top3 \* from employee\_table

order by salary desc;

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

create view sales\_olap\_v as

select product\_id,sale\_date,daily\_sales

,sum(daily\_sales) over(order by) daily\_sales

row unbounded preceding) as "csum"

from sales\_table;

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

create view empl\_200\_v as

select employee\_no as emp\_no

,last\_name as last

,salary/12 (format,'$$$$,$$9.99') as month\_salary

from employee\_table

where dept\_no = 200;

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

create view customer\_order\_v as

select customer\_name as customer

,order\_number

,order\_total (format '$$$,$$9.99') as total\_amount

from customer\_table as cust

,order\_table as ord

where cust.customer\_number=ord.customer\_number;

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

create view aggreg\_order\_v as

select customer\_number

,order\_date/100+19000

,count(order\_total)

,sum(order\_total)

,avg(order\_total)

from order\_table

group by customer\_number.yr\_mth\_orders;

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

create view emp\_hr\_v8 as select \* from employee\_table8;

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

update emp\_hr\_v8

set salary = 888888.88

where employee\_no = 2000000000;

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

select \* from employee\_table8

where employee\_no = 2000000;

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

**LAB 5**

**WHERE CLAUSE**

1.select first\_name,last\_name,class\_code,grade\_pt

from student\_table where first\_name='henry'

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

2.select first\_name as fname,

last\_name lname,

class\_code "class\_code",

grade\_pt as "avg",

student\_id

from student\_table where first\_name='henry'

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

3.select first\_name as fname,

last\_name lname,

class\_code "class\_code",

grade\_pt as "avg",

student\_id

from student\_table where fname='henry'

order by "avg"

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

4.select \*

from student\_table where

grade\_pt=0.00

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

5.select \*

from student\_table where

class\_code is NULL

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

6.select \*

from student\_table where

class\_code is not NULL

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

7.select \*

from student\_table where

grade\_pt>=3.0

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

8.select \*

from student\_table where

grade\_pt GE 3.0

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

9.select \*

from student\_table where

class\_code='FR'

and first\_name='henry'

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

10.select \*

from student\_table where

grade\_pt=3.0 or grade\_pt=4.0

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

11.select \*

from student\_table where

grade\_pt=3.0 and class\_code='SR'

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

12.select \*

from student\_table where

grade\_pt=4.0 or grade\_pt=3.0 and class\_code='SR'

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

13.select \*

from student\_table where

(grade\_pt=4.0 or grade\_pt=3.0) and class\_code='SR'

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

14.select \*

from student\_table where

grade\_pt in (3.0,4.0) and class\_code='SR'

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

15.select \*

from student\_table where

grade\_pt in (2.0,3.0,4.0)

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

16.select \*

from student\_table where

grade\_pt not in (2.0,3.0,4.0)

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

17.select \*

from student\_table where

grade\_pt not in (2.0,3.0,4.0)

or grade\_pt is null

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

18.select \*

from student\_table where

grade\_pt = any (2.0,3.0,4.0)

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

19.select \*

from student\_table where

grade\_pt NOT= ALL (2.0,3.0,4.0)

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

20.select \*

from student\_table where

grade\_pt between 2.0 and 4.0

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

21.select \*

from student\_table where

last\_name between 'L' and 'LZ'

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

22.select \*

from student\_table where

last\_name LIKE 'SM%'

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

23..select \*

from student\_table where

last\_name LIKE '\_a%'

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

24.select \*

from student\_table where

last\_name LIKE ALL ('%M%','%S%')

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

25.select \*

from student\_table where

last\_name LIKE ANY ('%M%','%S%')

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

26.select \*

from student\_table where

last\_name LIKE ALL ('%S%','%m%')

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

27.select \*

from student\_table where

first\_name like '%y'

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

28.select \* from student\_table where last\_name like '%n'

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

29.select \* from student\_table where trim ( last\_name) like '%n'

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

30.select \* from student\_table

where student\_id is null

and last\_name is null

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

31.select empno,first\_name from employee\_table

where empno = 2000000

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

32.select student\_id, last\_name from student\_table

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

33.select last\_name from student\_table where trim(last\_name) like '%n'

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

34.select \* from student\_table where first\_name like 'S@%' escape '@'

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

35.select \* from student\_table where TRIM(last\_name) LIKE 'T@\_' escape '@'

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

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

**SAMPLE FUNCTIONS**

1.select \* from student\_course\_table SAMPLE .25,.25

order by 1,2

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

2.select student\_id,course\_id ,SAMPLEID from student\_course\_table SAMPLE 5,5,5 order by 3,1,2

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

3.select student\_id,course\_id ,SAMPLEID from student\_course\_table SAMPLE with REPLACEMENT 5,5,5 order by 3,1,2

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

4.select student\_id,course\_id ,SAMPLEID from student\_course\_table SAMPLE .1,.1,.1 order by SAMPLEID

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

5.select student\_id,course\_id ,SAMPLEID from student\_course\_table SAMPLE RANDOMIZED ALLOCATION .1,.1,.1,.1

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

6.select student\_id,course\_id ,SAMPLEID from student\_course\_table SAMPLE RANDOMIZED ALLOCATION

when course\_id>200 then .1,.1 else .2,.2

end

order by 3

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

7.select count(distinct(course\_id))

from (sel course\_id from student\_course\_table sample 5) DT

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

8.select \* from sales\_table

where random(1,100)=5

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

9.select product\_id , count(daily\_sales) from sales\_table

group by 1

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

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

**SET OPERATORS FUNCTION**

1.select \* from table\_red

intersect

select \* from table\_blue

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

2.select \* from table\_red

union

select \* from table\_blue

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

3.select \* from table\_red

union all

select \* from table\_blue

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

4.select \* from table\_red

except

select \* from table\_blue

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

5.select \* from table\_red

minus

select \* from table\_blue

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

6.select \* from table\_blue

minus

select \* from table\_red

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

7.select \* from table\_blue

except

select \* from table\_red

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

8.select \* from table\_blue

minus

select \* from table\_red

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

9.select dept\_no,empno from employee\_table

intersect

select dept\_no from department\_table

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

10.select first\_name from employee\_table

intersect

select department\_name from department\_table

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

11.select dept\_no as Depty, empno as "The Mgr'" from employee\_table

intersect

select dept\_no from department\_table

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

12.select empno as manager, trim(last\_name) || ',' || first\_name as "name"

from employee\_table

inner join

(select empno from employee\_table intersect

select mgr\_no from department\_table) as teratom (empno)

on empno=empno

order by "name"

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

13.select empno, first\_name,last\_name,salary from

hierarchy\_table

union all

select empno,first\_name,last\_name,salary from employee\_table

group by 1,2,3,4

order by 2;

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

14.select department\_name,dept\_no from department\_table

union all

select department\_name,dept\_no from department\_table order by 1

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

15.select

'employee' (TITLE ' '),first\_name ||' ' || last\_name as "name"

from employee\_table

union all

select dept\_no ,'department',department\_name

from department\_table

order by 1,2

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

16.select

dept\_no,empno as empnoo from employee\_table

union all

select dept\_no, employee\_no from employee\_table

intersect all

select dept\_no from department\_table

minus

select dept\_no from department\_table

where

department\_name like '%sales%'

order by 1,2

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

17.insert into combined\_custs

sel \* from cust\_table\_east

union all

sel \* from cust\_table\_west

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

18.select product\_id as prod\_id, null as yr,null as mth,sum(daily\_sales)

from sales\_table

group by 1,2,3

union

select null

,null

,extract(year from sale\_date)

from sales\_table

group by 1,2,3

union

select null,

null,

extract(month from sale\_date)

,sum(daily\_sales)

from sales\_table

group by 1,2,3

order by 1 desc , 2,3

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

19.select product\_id as prod\_id, extract( year from sale\_date) as yr,

extract (month from sale\_date) as mth,

sum(daily\_sales ) as "total" from sales\_table

group by 1,2,3

union

sel product\_id,

extract(year from sale\_date)

,null

,sum(daily\_sales) from sales\_table

group by 1,2,3

union

select product\_id,

null,

null,

sum(daily\_sales) from sales\_table

group by 1,2,3

union

select null,null,null,

sum(daily\_sales) from sales\_table

group by 1,2,3

order by 1 desc, 2,3

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

20.sel productJD as prod JD

,extract(year from sale\_date) as yr

,extract(monthj from sale\_date) as mth

,sum(daily\_sales) as "total"

from sales\_table

group by 1,2,3

union

sel productJD

,null

,extract(year from sale\_date)

,sum(daily\_sales) from sales\_table

group by 1,2,3

union

select productJD

,NULL

,NULL

,sum(daily\_sales) from sales\_table

group by 1,2,3

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

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

**STORED PROCEDURE**

1.create procedure First\_procedure()

BEGIN

INSERT INTO CUstomer\_Table DEFAULT VALUES;

END:

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

2.call First\_procedure();

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

3.select \* from customer\_table order by 1;

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

4.create procedure Secondt\_procedure()

BEGIN

INSERT INTO CUstomer\_Table DEFAULT VALUES;

secondsection:BEGIN

Delete from customer\_table where customer\_number is null;

END secondsection;

END;

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

5.call second\_procedure();

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

6.create procedure declare\_procedure()

BEGIN

DECLARE var1 INTEGER DEFAULT 11111111;

Delete from Customer\_Table where customer\_number=:var1;

END;

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

7.call Declare\_procedure()

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

8.create procedure SetVar\_procedure()

BEGIN

DECLARE var1 INTEGER ;

set var1=31313131;

Delete from Customer\_Table where customer\_number=:var1;

END;

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

9.call SetVar\_procedure()

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

10.create procedure PassInput\_procedure(IN var1 INTEGER)

BEGIN

Delete from customer\_table where customer\_number=:var1l;

END;

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

11.call PassInput\_procedure(31313131)

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

12.create procedure Test\_proc (IN var1 BYTEINT,IN var2 BYTEINT,OUT msg CHAR(20))

BEGIN

CASE WHEN var1=var2 then set msg='They are equal';

WHEN var1<var2 then set msg='variable1 is less';

ELSE set msg='Variable1 is greater';

END CASE;

END;

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

13.call Test\_Proc(1,2,Msg);

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

14.create procedure TestIF\_proc

(IN var1 BYTEINT,IN var2 BYTEINT,OUT msg CHAR(20))

BEGIN

IF var1=var2 then set msg='They are equal';

END IF;

IF var1<var2 then set msg='variable1 is less';

END IF;

IF var1>var2 then set msg='variable1 is graeter';

END IF;

END;

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

15.CREATE PROCEDURE Inserter\_Five()

LOOPER:BEGIN

DECLARE cntr INTEGER DEFAULT 0;

loopit:LOOP

SET cntr=cntr+1;

IF cntr>5 THEN LEAVE loopit;

END IF;

Insert into my\_log\_tb1 VALUES(:cntr,Time);

END LOOP loopit;

END LOOPER;

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

16.call Inserter\_Five();

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

17.select \* from my\_log\_tb1 order by 1;

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

18.CREATE PROCEDURE Ins5()

LOOPER:BEGIN

DECLARE cntr INTEGER DEFAULT 0;

loopit:LOOP

SET cntr=cntr+1;

IF cntr>5 THEN LEAVE loopit;

END IF;

Insert into my\_log\_tb1 VALUES(:cntr,Time);

END LOOP loopit;

END LOOPER;

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

19.CREATE PROCEDURE Ins5()

LOOPER:BEGIN

DECLARE cntr INTEGER DEFAULT 0;

loopit:REPEAT

SET cntr=cntr+1;

Insert into my\_log\_tb1 VALUES(:cntr,Time);

until cntr>4

END REPEAT loopit;

END LOOPER;

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

20.create multiset table sql01.InsProcXYZ

( col1 INTEGER

,col2 INTEGER)

primary index(col1);

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

21.create procedure sql01.InsProcTLC()

BEGIN

DECLARE mynumber INTEGER default 0;

set mynumber=1000

myloop:loop

set mynumber=mynumber+1;

if mynumber>2000 then leave myloop;

end if;

insert into sql01.InsProcXYZ

(:mynumber,:mynumber MOD 250);

END LOOP myloop;

END;

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

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

**SQL QUERY FUNCTION**

1.select \* from employee\_table

where deptno in(100,200);

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

2.select \* from employee\_table

where deptno in(100,100,200,200);

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

3.select \* from employee\_table

where deptno in(select dept\_no from department\_table);

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

4.select \* from employee\_table

where deptno in(100,200,300,400,500);

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

5.select \* from employee\_table as e

inner join

department\_table as d

on e.deptno=d.dept\_no;

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

6.select \* from customer\_table

where customer\_number in(select customer\_number from order\_table);

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

7.select \* from customer\_table

where customer\_number in(select customer\_number from order\_table

where order\_total>100000.00);

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

8.select \* from employee\_table where salary>(select avg(salary) from employee\_table);

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

9.select \* from employee\_table as e

where salary>

(select avg(salary) from employee\_table as ee

where e.deptno=ee.deptno);

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

10.select last\_name,deptno,salary from employee\_table as e

where salary>

(select avg(salary) from employee\_table as ee

where e.deptno=ee.deptno);

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

11.select e.\*,avgsal from employee\_table as e

inner join

(select deptno,avg(salary) from employee\_table

group by deptno)

as TeraTom(Depty,avgsal)

on deptno=depty

and salary>avgsal;

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

12.select \* from sales\_table as Tops

where Daily\_sales>(

select avg(Daily\_sales)

from sales\_table as Bots

where Tops.product\_id=Bots.product\_id)

order by product\_id,sale\_date;

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

13.select \* from sales\_table as Tops

where Daily\_sales>(

select avg(Daily\_sales)

from sales\_table as Bots

where Tops.sale\_date=Bots.sale\_date)

order by sale\_date;

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

14.select \* from student\_table as Tops

where Grade\_pt>(

select avg(Grade\_pt)

from student\_table as Bots

where Tops.class\_code=Bots.class\_code)

order by class\_code

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