BASIC QUERIES

DDL COMMANDS

CREATE

To create tables, views, synonyms, sequences, functions, procedures, packages etc.

```
SQL>create table emp (empno number(5),
        name varchar2(20),
        sal number(10,2),
        job varchar2(20),
        mgr  number(5),
        Hiredate  date,
        comm number(10,2));
```

Table Created

CREATE AS

SQL>create table emp2 as select * from emp; // To create the copies of tables

Table Created

ALTER

Use the ALTER TABLE statement to alter the structure of a table.

```
SQL>alter table emp add (addr varchar2(20), city varchar2(20),
   pin varchar2(10),ph varchar2(20));
```

Table Altered.

DROP

TO DROP COLUMNS

SQL>alter table emp drop column (pin, city);

Table Altered

TO DROP TABLE

SQL>drop table emp;

Table Dropped.

RENAME

Use the RENAME statement to rename a table, view, sequence, or private synonym for a table, view, or sequence.

SQL>rename emp2 to employee2;

Table Renamed.

TRUNCATE

Use the Truncate statement to delete all the rows from table permanently.

SQL> truncate table emp;

DML COMMANDS

SELECT – retrieve data from the a database
INSERT – insert data into a table
UPDATE – updates existing data within a table
DELETE – deletes all records from a table, the space for the records remain
MERGE – UPSERT operation (insert or update)
CALL – call a PL/SQL or Java subprogram
LOCK TABLE – control concurrency

INSERT

SYNTAX

INSERT INTO <table name> VALUES (<value 1>, ... <value n>);

EXAMPLE:
SQL>INSERT INTO STUDENT VALUES (1001,'Ram');

Record Inserted.

UPDATE :

The update statement is used to change values that are already in a table.

SYNTAX

UPDATE <table name> SET <attribute> = <expression> WHERE <condition>;

Example:

SQL>UPDATE STUDENT SET Name = 'Amar' WHERE StudID=1001;

Table Updated.

SELECT :

The SELECT statement is used to form queries for extracting information out of the database.

SELECT <attribute>, ….., <attribute n> FROM <table name>;

Example:

SQL>SELECT StudID, Name FROM STUDENT;

OPERATORS, FUNCTIONS, EXPRESSIONS, CONDITIONS

ARITHMETIC OPERATORS

Use an arithmetic operator in an expression to negate, add, subtract, multiply, and divide numeric values. The result of the operation is also a numeric value.

Operator	Purpose	Example
+ -	Denotes a positive or negative expression. These are unary operators.	SELECT * FROM orders WHERE qtysold = -1; SELECT * FROM emp WHERE -sal < 0;
* /	Multiplies, divides. These are binary operators.	UPDATE emp SET sal = sal * 1.1;
+ -	Adds, subtracts. These are binary operators.	SELECT sal + comm FROM emp WHERE SYSDATE - hiredate > 365;

CONCATENATION OPERATOR

The concatenation operator manipulates character strings

Operator	Purpose	Example
||	Concatenates character strings.	SELECT 'Name is ' || ename FROM emp;

COMPARISON OPERATORS

Comparison operators compare one expression with another. The result of such a comparison can be TRUE, FALSE, or UNKNOWN.

Operator	Purpose	Example
=	Equality test.	SELECT * FROM emp WHERE sal = 1500;
!= ^= <> ¬=	Inequality test. Some forms of the inequality operator may be unavailable on some platforms.	SELECT * FROM emp WHERE sal != 1500;
> <	"Greater than" and "less than" tests.	SELECT * FROM emp WHERE sal > 1500; SELECT * FROM emp WHERE sal < 1500;
>= <=	"Greater than or equal to" and "less than or equal to" tests.	SELECT * FROM emp WHERE sal >= 1500; SELECT * FROM emp WHERE sal <= 1500;
IN	"Equal to any member of" test. Equivalent to "= ANY".	SELECT * FROM emp WHERE job IN ('CLERK','ANALYST'); SELECT * FROM emp WHERE sal IN

		(SELECT sal FROM emp WHERE deptno = 30);
NOT IN	Equivalent to "!=ALL". Evaluates to FALSE if any member of the set is NULL.	SELECT * FROM emp WHERE sal NOT IN (SELECT sal FROM emp WHERE deptno = 30); SELECT * FROM emp WHERE job NOT IN ('CLERK', ANALYST');
ANY SOME	Compares a value to each value in a list or returned by a query. Must be preceded by =, !=, >, <, <=, >=. Evaluates to FALSE if the query returns no rows.	SELECT * FROM emp WHERE sal = ANY (SELECT sal FROM emp WHERE deptno = 30);
ALL	Compares a value to every value in a list or returned by a query. Must be preceded by =, !=, >, <, <=, >=. Evaluates to TRUE if the query returns no rows.	SELECT * FROM emp WHERE sal >= ALL (1400, 3000);
[NOT] BETWEEN x AND y	[Not] greater than or equal to *x* and less than or equal to *y*.	SELECT * FROM emp WHERE sal BETWEEN 2000 AND 3000;
EXISTS	TRUE if a subquery returns at least one row.	SELECT ename, deptno FROM dept WHERE EXISTS (SELECT * FROM emp WHERE dept.deptno = emp.deptno);
x [NOT] LIKE y [ESCAPE 'z']	TRUE if *x* does [not] match the pattern *y*. Within *y*, the character "%" matches any string of zero or more characters except null. The character "_" matches any single character. Any character, excepting percent (%) and underbar (_) may follow ESCAPE; a wildcard character is treated as a literal if preceded by the character designated as the	See "LIKE Operator" SELECT * FROM tab1 WHERE col1 LIKE 'A_C/%E%' ESCAPE '/';

	escape character.	
IS [NOT] NULL	Tests for nulls. This is the only operator that you should use to test for nulls. See "Nulls".	SELECT ename, deptno FROM emp WHERE comm IS NULL;

NOT IN Operator

If any item in the list following a NOT IN operation is null, all rows evaluate to UNKNOWN (and no rows are returned). For example, the following statement returns the string 'TRUE' for each row:

SQL>SELECT 'TRUE' FROM emp WHERE deptno NOT IN (5,15);

LIKE OPERATOR

The LIKE operator is used in character string comparisons with pattern matching.

SQL>SELECT sal FROM emp WHERE ename LIKE 'SM%';

The following query uses the = operator, rather than the LIKE operator, to find the salaries of all employees with the name 'SM%':

SQL>SELECT sal FROM emp WHERE ename = 'SM%';

LOGICAL OPERATORS

A logical operator combines the results of two component conditions to produce a single result based on them or to invert the result of a single condition.

Operator	Function	Example
NOT	Returns TRUE if the following condition is FALSE. Returns FALSE if it is TRUE. If it is UNKNOWN, it remains UNKNOWN.	SELECT * FROM emp WHERE NOT (job IS NULL); SELECT * FROM emp WHERE NOT (sal BETWEEN 1000 AND 2000);
AND	Returns TRUE if both component conditions are TRUE. Returns FALSE if	SELECT * FROM emp

	either is FALSE. Otherwise returns UNKNOWN.	WHERE job = 'CLERK' AND deptno = 10;
OR	Returns TRUE if either component condition is TRUE. Returns FALSE if both are FALSE. Otherwise returns UNKNOWN.	SELECT * FROM emp WHERE job = 'CLERK' OR deptno = 10;

SET OPERATORS

Operator	Returns
UNION	All rows selected by either query.
UNION ALL	All rows selected by either query, including all duplicates.
INTERSECT	All distinct rows selected by both queries.
MINUS	All distinct rows selected by the first query but not the second.

UNION Example

SQL>SELECT part, partnum, to_date(null) date_in FROM orders_list1UNION
SELECT part, to_null(null), date_inFROM orders_list2;

```
PART      PARTNUM DATE_IN
---------- ------- --------
SPARKPLUG  3323165
SPARKPLUG        10/24/98
FUEL PUMP  3323162
FUEL PUMP        12/24/99
TAILPIPE   1332999
TAILPIPE         01/01/01
CRANKSHAFT 9394991
CRANKSHAFT       09/12/02
```

SQL>SELECT part FROM orders_list1 UNION SELECT part FROM orders_list2;

```
PART
----------
SPARKPLUG
FUEL PUMP
TAILPIPE
```

CRANKSHAFT

UNION ALL Example

SQL>SELECT part FROM orders_list1 UNION ALL SELECT part FROM orders_list2;

```
PART
----------
SPARKPLUG
FUEL PUMP
FUEL PUMP
TAILPIPE
CRANKSHAFT
TAILPIPE
TAILPIPE
```

INTERSECT Example

SQL>SELECT part FROM orders_list1 INTERSECT SELECT part FROM orders_list2;

```
PART
----------
TAILPIPE
```

MINUS Example

SQL>SELECT part FROM orders_list1 MINUS SELECT part FROM orders_list2;

```
PART
----------
SPARKPLUG
FUEL PUMP
```

FUNCTIONS

Functions are of these general types:

- single-row (or scalar) functions
- group (or aggregate) functions

NUMBER FUNCTIONS

ABS

Example SQL>SELECT ABS(-15) "Absolute" FROM DUAL;

```
 Absolute
----------
       15
```

ACOS

Example SQL>SELECT ACOS(.3)"Arc_Cosine" FROM DUAL;

```
Arc_Cosine
----------
1.26610367
```

ASIN

Example SQL>SELECT ASIN(.3) "Arc_Sine" FROM DUAL;

```
 Arc_Sine
----------
.304692654
```

ATAN

Example SQL>SELECT ATAN(.3) "Arc_Tangent" FROM DUAL;

```
Arc_Tangent
----------
.291456794
```

CEIL

Example SQL>SELECT CEIL(15.7) "Ceiling" FROM DUAL;

```
  Ceiling
----------
       16
```

EXP

Purpose Returns e raised to the *n*th power; e = 2.71828183 ...

Example SQL>SELECT EXP(4) "e to the 4th power" FROM DUAL;

```
e to the 4th power
```

```
------------------
       54.59815
```

FLOOR

Example SQL>SELECT FLOOR(15.7) "Floor" FROM DUAL;

```
     Floor
----------
        15
```

LOG

Example SQL>SELECT LOG(10,100) "Log base 10 of 100" FROM DUAL;

```
Log base 10 of 100
------------------
                 2
```

MOD

Syntax MOD(m,n)

Example SQL>SELECT MOD(11,4) "Modulus" FROM DUAL;

```
   Modulus
----------
         3
```

POWER

Example SQL>SELECT POWER(3,2) "Raised" FROM DUAL;

```
    Raised
----------
         9
```

ROUND

Syntax ROUND(n[,m])

Example 1 SQL>SELECT ROUND(15.193,1) "Round" FROM DUAL;

```
     Round
```

```
----------
    15.2
```

Example 2 SQL>SELECT ROUND(15.193,-1) "Round" FROM DUAL;

```
     Round
----------
      20
```

SIGN

Syntax SIGN(n)

Example SQL>SELECT SIGN(-15) "Sign" FROM DUAL;

```
      Sign
----------
        -1
```

SIN

Example SQL>SELECT SIN(30 * 3.14159265359/180)"Sine of 30 degrees" FROM DUAL;

```
Sine of 30 degrees
------------------
                .5
```

SQRT

Example SQL>SELECT SQRT(26) "Square root" FROM DUAL;

```
Square root
-----------

5.09901951
```

TAN

Example SQL>SELECT TAN(135 * 3.14159265359/180)"Tangent of 135 degrees" FROM DUAL;

```
Tangent of 135 degrees
----------------------
```

CHARACTER FUNCTIONS

CHR

Syntax CHR(n [USING NCHAR_CS])

CONCAT

Syntax CONCAT(char1, char2)

Example
```
SQL>SELECT CONCAT( CONCAT(ename, ' is a '), job) "Job"
FROM empWHERE empno = 7900;

Job
-----------------
JAMES is a CLERK
```

INITCAP

Example
```
SQL>SELECT INITCAP('the soap') "Capitals" FROM DUAL;

Capitals
---------
The Soap
```

LOWER

Example
```
SQL>SELECT LOWER('MR. SCOTT MCMILLAN') "Lowercase"   FROM DUAL;

Lowercase
--------------------
mr. scott mcmillan
```

REPLACE

Syntax REPLACE(char,search_string[,replacement_string])

Example
```
SQL>SELECT REPLACE('JACK and JUE','J','BL') "Changes" FROM DUAL;

Changes
```

```
--------------
BLACK and BLUE
```

UPPER

Syntax UPPER(char)

Example SQL>SELECT UPPER('Large') "Uppercase" FROM DUAL;

```
Upper
-----
LARGE
```

DATE FUNCTIONS

MONTHS_BETWEEN

Syntax MONTHS_BETWEEN(d1, d2)

Example

```
SQL>SELECT MONTHS_BETWEEN
(TO_DATE('02-02-1995','MM-DD-YYYY'),
 TO_DATE('01-01-1995','MM-DD-YYYY') ) "Months"
 FROM DUAL;

  Months
----------
1.03225806
```

ROUND

Syntax ROUND(d[,fmt])

Example SQL>SELECT ROUND (TO_DATE ('27-OCT-18'),'YEAR') "New Year" FROM DUAL;

```
New Year
---------
01-JAN-19
```

SYSDATE

Syntax SYSDATE

Example SELECT TO_CHAR(SYSDATE, 'MM-DD-YYYY HH24:MI:SS')"NOW" FROM DUAL;

```
NOW
-------------------
10-29-2019 20:27:11
```

GROUP FUNCTIONS

AVG

Syntax AVG([DISTINCT|ALL] n)

Example SQL>SELECT AVG(sal) "Average"FROM emp;

```
   Average
----------
2077.21429
```

COUNT

Syntax COUNT({* | [DISTINCT|ALL] expr})

Example 1 SQL>SELECT COUNT(*) "Total" FROM emp;

```
   Total
----------
      18
```

Example 2 SQL>SELECT COUNT(job) "Count" FROM emp;

```
Count
----------
      14
```

Example 3 SQL>SELECT COUNT(DISTINCT job) "Jobs" FROM emp;

```
     Jobs
----------
       5
```

MAX

Syntax MAX([DISTINCT|ALL] expr)

Example SQL>SELECT MAX(sal) "Maximum" FROM emp;

```
   Maximum
----------
      5000
```

MIN

Syntax MIN([DISTINCT|ALL] expr)

Example SQL>SELECT MIN(hiredate) "Earliest" FROM emp;

```
Earliest
---------
17-DEC-80
```

SUM

Syntax SUM([DISTINCT|ALL] n)

Example SQL>SELECT SUM(sal) "Total"FROM emp;

```
    Total
----------
    29081
```

Ex. No: 1

Date:

CREATION OF CLIENT MASTER TABLE WITH THE REQUIRED FIELDS USING SQL QUERIES

Aim:

To create the Client_Master table with the required fields and processing the instructions using SQL Queries.

QUERIES:

Table Creation Client Master

```
SQL> create table client_master(client_no number,
  2  client_name varchar2(10),
  3  address varchar2(15),
  4  city varchar2(15),
  5  state varchar2(15),
  6  pincode number(6),
  7  remark varchar2(10),
  8  bal_due number(5));
```

Table created.

i) Table Creation Supplier using as

```
SQL> create table supplier as(select *from client_master);
```

Table created.

SQL> desc client_master;

Name	Null?	Type
CLIENT_NO		NUMBER
CLIENT_NAME		VARCHAR2(10)
ADDRESS		VARCHAR2(15)
CITY		VARCHAR2(15)
STATE		VARCHAR2(15)
PINCODE		NUMBER(6)
REMARK		VARCHAR2(10)
BAL_DUE		NUMBER(5)

I. Table alteration_client_no,name to supplier_no,name using rename

SQL> alter table supplier rename column client_no to supplier_no;

Table altered.

SQL> alter table supplier rename column client_name to supplier_name;

Table altered.

SQL> desc supplier;

Name	Null?	Type
SUPPLIER_NO		NUMBER

SUPPLIER_NAME	VARCHAR2(10)
ADDRESS	VARCHAR2(15)
CITY	VARCHAR2(15)
STATE	VARCHAR2(15)
PINCODE	NUMBER(6)
REMARK	VARCHAR2(10)
BAL_DUE	NUMBER(5)

ii) Insert the values into Client_Master

```
SQL>insertinto client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due);

Enter value for client_no: 101

Enter value for client_name: abinaya

Enter value for address: panaiyur

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623530

Enter value for remark: paid

Enter value for bal_due: 0

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(101,'abinaya','panaiyur','ramnad','tamilnadu',623530,'paid',0)

1 row created.
```

SQL> /

Enter value for client_no: 102

Enter value for client_name: arthi

Enter value for address: aranmanai

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623501

Enter value for remark: notpaid

Enter value for bal_due: 1000

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(102,'arthi','aranmanai','ramnad','tamilnadu',623501,'notpaid',1000)

1 row created.

SQL> /

Enter value for client_no: 103

Enter value for client_name: kalai

Enter value for address: akasthiyarkutam

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623574

Enter value for remark: paid

Enter value for bal_due: 0

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(103,'kalai','akasthiyarkutam','ramnad','tamilnadu',623574,'paid',0)

1 row created.

SQL> /

Enter value for client_no: 104

Enter value for client_name: kavya

Enter value for address: semanoor

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623507

Enter value for remark: notpaid

Enter value for bal_due: 5000

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(104,'kavya','semanoor','ramnad','tamilnadu',623507,'notpaid',5000)

1 row created.

SQL> /

Enter value for client_no: 105

Enter value for client_name: natheeba

Enter value for address: pattinamkathan

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623704

Enter value for remark: paid

Enter value for bal_due: 0

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(105,'natheeba','pattinamkathan','ramnad','tamilnadu',623704,'paid',0)

1 row created.

SQL> /

Enter value for client_no: 106

Enter value for client_name: rani

Enter value for address: semanoor

Enter value for city: ramnad

Enter value for state: tamilnadu

Enter value for pincode: 623506

Enter value for remark: notpaid

Enter value for bal_due: 9000

old 1: insert into client_master values(&client_no,'&client_name','&address','&city','&state',&pincode,'&remark',&bal_due)

new 1: insert into client_master values(106,'rani','semanoor','ramnad','tamilnadu',623506,'notpaid',9000)

1 row created.

SQL> select *from client_master;

CLIENT_NO	CLIENT_NAME	ADDRESS	CITY	STATE	PINCODE	REMARK	BAL_DUE
101	abinaya	panaiyur	ramnad	tamilnadu	623530	paid	0
102	arthi	aranmanai	ramnad	tamilnadu	623501	notpaid	1000
103	kalai	akasthiyarkutam	ramnad	tamilnadu	623574	paid	0
104	kavya	semanoor	ramnad	tamilnadu	623507	notpaid	5000
105	natheeba	pattinamkathan	ramnad	tamilnadu	623704	paid	0
106	rani	semanoor	ramnad	tamilnadu	623506	notpaid	9000

6 rows selected.

iii) Insert Data into Supplier from Client_Master

SQL> insert into supplier(select *from client_master);

6 rows created.

SQL> select *from supplier;

SUPPLIER_NO	SUPPLIER_NA	ADDRESS	CITY	STATE	PINCODE	REMARK	BAL_DUE
101	abinaya	panaiyur	ramnad	tamilnadu	623530	paid	0
102	arthi	aranmanai	ramnad	tamilnadu	623501	notpaid	1000
103	kalai	akasthiyarkutam	ramnad	tamilnadu	623574	paid	0
104	kavya	semanoor	ramnad	tamilnadu	623507	notpaid	5000
105	natheeba	pattinamkathan	ramnad	tamilnadu	623704	paid	0
106	rani	semanoor	ramnad	tamilnadu	623506	notpaid	9000

6 rows selected.

iv) Delete the selected row in the Client_Master

SQL> delete from client_master where client_no=106;

1 row deleted.

SQL> select *from client_master;

CLIENT_NO	CLIENT_NAME	ADDRESS	CITY	STATE	PINCODE	REMARK	BAL_DUE
101	abinaya	panaiyur	ramnad	tamilnadu	623530	paid	0
102	arthi	aranmanai	ramnad	tamilnadu	623501	notpaid	1000
103	kalai	akasthiyarkutam	ramnad	tamilnadu	623574	paid	0
104	kavya	semanoor	ramnad	tamilnadu	623507	notpaid	5000
105	natheeba	pattinamkathan	ramnad	tamilnadu	623704	paid	0

5 rows selected.

Ex. No: 2

Date:

CREATION OF SALES_ORDER TABLE WITH THE REQUIRED FIELDS BY CONSTRAINTS USING SQL QUERIES

Aim:

To create the Sales_Order table with the required fields by constraints and processing the instructions using SQL Queries.

QUERIES:

a) Table Creation_sales_order

```
SQL> create table sales_order(s_order_no number(5) primary key,
  2  client_no number(5),
  3  delivery_address varchar2(15),
  4  delivery_date date,
  5  order_status varchar2(15));

Table created.

SQL> desc sales_order;
 Name                                      Null?                Type
 ----------------------------------------- -------- ----------------------------
 S_ORDER_NO                                NOT NULL NUMBER(5)
 CLIENT_NO                                          NUMBER(5)
 DELIVERY_ADDRESS                          VARCHAR2(15)
 DELIVERY_DATE                                      DATE
 ORDER_STATUS                                       VARCHAR2(15)
```

b) Table alteration

SQL> alter table sales_order add salesman_no number(5);

Table altered.

SQL> desc sales_order;

Name	Null?	Type
S_ORDER_NO	NOT NULL	NUMBER(5)
CLIENT_NO		NUMBER(5)
DELIVERY_ADDRESS		VARCHAR2(15)
DELIVERY_DATE		DATE
ORDER_STATUS		VARCHAR2(15)
SALESMAN_NO		NUMBER(5)

SQL> insert into sales_order values(&s_order_no,&client_no,'&delivery_address', '&delivery_date','&order_status',&salesman_no);

Enter value for s_order_no: 101

Enter value for client_no: 200

Enter value for delivery_address: north street

Enter value for delivery_date: 10_jan_18

Enter value for order_status: delivered

Enter value for salesman_no: 1001

old 1: insert into sales_order values(&s_order_no,&client_no,'&delivery_addre ss','&delivery_date','&order_status',&salesman_no)

new 1: insert into sales_order values(101,200,'north street','10_jan_18','del ivered',1001)

```
1 row created.
SQL> /
Enter value for s_order_no: 102
Enter value for client_no: 200
Enter value for delivery_address: north street
Enter value for delivery_date: 30_jun_19
Enter value for order_status: not delivered
Enter value for salesman_no: 1002
old   1: insert into sales_order values(&s_order_no,&client_no,'&delivery_addre
ss','&delivery_date','&order_status',&salesman_no)
new   1: insert into sales_order values(102,200,'north street','30_jun_19','not
 delivered',1002)
1 row created.

SQL> /
Enter value for s_order_no: 103
Enter value for client_no: 201
Enter value for delivery_address: ramnad
Enter value for delivery_date: 18_jun_19
Enter value for order_status: pending
Enter value for salesman_no: 1002
old   1: insert into sales_order values(&s_order_no,&client_no,'&delivery_addre
ss','&delivery_date','&order_status',&salesman_no)
new   1: insert into sales_order values(103,201,'ramnad','18_jun_19','pending',
1002)
```

1 row created.

SQL> /

Enter value for s_order_no: 104

Enter value for client_no: 201

Enter value for delivery_address: ramnad

Enter value for delivery_date: 21_jun_19

Enter value for order_status: delivered

Enter value for salesman_no: 1003

old 1: insert into sales_order1 values(&s_order_no,&client_no,'&delivery_addre
ss','&delivery_date','&order_status',&salesman_no)

new 1: insert into sales_order1 values(104,201,'ramnad','21_jun_19','delivered
',1003)

1 row created.

SQL> /

Enter value for s_order_no: 105

Enter value for client_no: 205

Enter value for delivery_address: aranmanai

Enter value for delivery_date: 01_aug_19

Enter value for order_status: not delivered

Enter value for salesman_no: 1004

old 1: insert into sales_order values(&s_order_no,&client_no,'&delivery_addre
ss','&delivery_date','&order_status',&salesman_no)

new 1: insert into sales_order values(105,205,'aranmanai','01_aug_19','not de
livered',1004)

1 row created.

SQL> select *from sales_order;

S_ORDER_NO	CLIENT_NO	DELIVERY_ADDRES	DELIVERY_DATE	ORDER_STATUS	SALESMAN_NO
101	200	north street	10-JAN-18	delivered	1001
102	200	north street	30-JUN-19	not delivered	1002
103	201	ramnad	18-JUN-19	pending	1002
104	201	ramnad	21-JUN-19	delivered	1003
105	205	aranmanai	01-AUG-19	not delivered	1004

I. **Table creation item_detail set as foreign key of s_order_no**.

SQL> create table item_detail(item_name varchar2(15),

2 s_order_no references sales_order(s_order_no));

Table created.

SQL> desc item_detail;

Name	Null?	Type
ITEM_NAME		VARCHAR2(15)
S_ORDER_NO		NUMBER(5)

SQL> insert into item_detail values('&item_name',&s_order_no);

Enter value for item_name: chocolate

Enter value for s_order_no: 101

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('chocolate',101)

1 row created.

SQL> /

Enter value for item_name: biscuits

Enter value for s_order_no: 104

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('buscuits',104)

1 row created.

SQL> /

Enter value for item_name: shampoo

Enter value for s_order_no: 101

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('shampoo',101)

1 row created.

SQL> /

Enter value for item_name: soap

Enter value for s_order_no: 102

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('soap',102)

1 row created.

SQL> /

Enter value for item_name: turmeric

Enter value for s_order_no: 103

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('turmeric',103)

1 row created.

SQL> /

Enter value for item_name: hairoil

Enter value for s_order_no: 103

old 1: insert into item_detail values('&item_name',&s_order_no)

new 1: insert into item_detail values('hairoil',103)

1 row created.

SQL> select *from item_detail;

ITEM_NAME	S_ORDER_NO
chocolate	101
biscuits	104
shampoo	101
soap	102
turmeric	103
hairoil	103

6 rows selected.

II. Integrity rules using check constraints.

SQL> alter table sales_order1 add check(s_order_no<150);

Table altered.

SQL> insert into sales_order1 values(160,206,'agraharam','01_sep_19','not delive red',1005);

```
insert into sales_order1 values(160,206,'agraharam','01_sep_19','not delivered',
1005)
*
ERROR at line 1:
ORA-02290: check constraint (SCOTT.SYS_C009718) violated


SQL> insert into item_detail values('olive oil',108);
insert into item_detail values('olive oil',108)
*
ERROR at line 1:
ORA-02291: integrity constraint (SCOTT.SYS_C009717) violated - parent key not
Found
```

Ex. No: 3

Date:

CREATION OF STUDENT MASTER TABLE WITH THE REQUIRED FIELDS USING SQL QUERIES

Aim:

To create the Student Master table with the required fields by constraints and processing the instructions using SQL Queries.

QUERIES:

I. Table Creation _Student_Master

```
SQL> create table student_master(name varchar2(10),
  2  regno number(10),
  3  dept varchar2(5),
  4  year number(4));
Table created.
SQL> desc student_master;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 NAME                                               VARCHAR2(10)
 REGNO                                              NUMBER(10)
 DEPT                                               VARCHAR2(5)
 YEAR                                               NUMBER(4)

SQL> insert into student_master values('&name',&regno,'&dept',&year);
Enter value for name: deepa
Enter value for regno: 111
```

Enter value for dept: cs

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('deepa',111,'cs',2019)

1 row created.

SQL> /

Enter value for name: rani

Enter value for regno: 112

Enter value for dept: tam

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('rani',112,'tam',2019)

1 row created.

SQL> /

Enter value for name: sheela

Enter value for regno: 113

Enter value for dept: cs

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('sheela',113,'cs',2019)

1 row created.

SQL> /

Enter value for name: sheela

Enter value for regno: 123

Enter value for dept: cs

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('sheela',113,'cs',2013)

1 row created.

SQL> /

Enter value for name: sreesha

Enter value for regno: 114

Enter value for dept: cs

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('sreesha',114,'cs',2019)

1 row created.

SQL> /

Enter value for name: rega

Enter value for regno: 115

Enter value for dept: eng

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('rega',115,'eng',2019)

1 row created.

SQL> /

Enter value for name: nila

Enter value for regno: 116

Enter value for dept: maths

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('nila',116,'maths',2019)

1 row created.

SQL> /

Enter value for name: banu

Enter value for regno: 117

Enter value for dept: maths

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('banu',117,'maths',2019)

1 row created.

SQL> /

Enter value for name: meena

Enter value for regno: 118

Enter value for dept: com

Enter value for year: 2019

old 1: insert into student_master values('&name',&regno,'&dept',&year)

new 1: insert into student_master values('meena',118,'com',2019)

1 row created.

SQL> select *from student_master;

NAME	REGNO	DEPT	YEAR
deepa	111	cs	2019
rani	112	tam	2019
sheela	113	cs	2019
sheela	123	cs	2019
sreesha	114	cs	2019
rega	115	eng	2019
nila	116	maths	2019
banu	117	maths	2019

meena 118 com 2019

9 rows selected.

a. select the student name column.

SQL> select name from student_master;

NAME

---

deepa

rani

sheela

sheela

sreesha

rega

nila

banu

meena

9 rows selected.

b.Eliminate the duplicate entry in table.

SQL>delete student_master where regno not in(select max(regno) from student_master group by name,dept);

1 row deleted.

SQL> select *from student_master;

NAME	REGNO	DEPT	YEAR
deepa	111	cs	2019
rani	112	tam	2019
sheela	113	cs	2019

sreesha	114	cs	2019
rega	115	eng	2019
nila	116	maths	2019
banu	117	maths	2019
meena	118	com	2019

9 rows selected.

c.sort the table in alphabetical order.

SQL> select *from student_master order by name;

NAME	REGNO	DEPT	YEAR
banu	117	maths	2019
deepa	111	cs	2019
meena	118	com	2019
nila	116	maths	2019
rani	112	tam	2019
rega	115	eng	2019
sheela	113	cs	2019
sreesha	114	cs	2019

8 rows selected.

d.Select all the students of a particular department.

SQL> select *from student_master where dept='cs';

NAME	REGNO	DEPT	YEAR
deepa	111	cs	2019
sheela	113	cs	2019
sreesha	114	cs	2019

Ex. No: 4

Date:

CREATION OF SALES_ORDER_DETAIL TABLE WITH THE REQUIRED FIELDS USING CONSTRAINTS BY SQL QUERIES

Aim:

To create the Sales_Order table with the required fieldsusing constraints and processing the instructions by SQL Queries.

QUERIES:

I. Table Creation _Sales_ord_detail.

```
SQL> create table sales_ord_detail(s_o_no number(5) primary key,
  2  product_no number(5),
 3 description varchar2(10),
  4  qty_ordered number(4,2),
  5  qty_disp number(4,2),
  6  product_rate number(4,2),
  7  profit_percent number(4,2),
  8  sell_price number(4,2),
  9  supplier_name varchar2(10));
Table created.
SQL> desc sales_ord_detail;
 Name                                      Null?                Type
 ----------------------------------------- -------------- ----------------------------
 S_O_NO                                    NOT NULL             NUMBER(5)
 PRODUCT_NO                                                     NUMBER(5)
 DESCRIPTION                                                    VARCHAR2(10)
 QTY_ORDERED                                                    NUMBER(4,2)
```

QTY_DISP	NUMBER(4,2)
PRODUCT_RATE	NUMBER(4,2)
PROFIT_PERCENT	NUMBER(4,2)
SELL_PRICE	NUMBER(4,2)
SUPPLIER_NAME	VARCHAR2(10)

```
SQL> insert into sales_ord_detail
     values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rat
     e,&profit_percent,&sell_price,'&supplier_name');

Enter value for s_o_no: 1234

Enter value for product_no: 1121

Enter value for description: shampoo

Enter value for qty_ordered: 20

Enter value for qty_disp: 10

Enter value for product_rate: 20.5

Enter value for profit_percent: 10

Enter value for sell_price: 30.5

Enter value for supplier_name: raj

old   1: insert into sales_ord_detail
     values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rat
     e,&profit_percent,&sell_price,'&supplier_name')

new   1: insert into sales_ord_detail values(1234,1121,'shampoo',20,10,20.5,10,30.5,'raj')

1 row created.

SQL> /

Enter value for s_o_no: 1235

Enter value for product_no: 1131

Enter value for description: lipstick

Enter value for qty_ordered: 10

Enter value for qty_disp: 15
```

Enter value for product_rate: 15

Enter value for profit_percent: 5

Enter value for sell_price: 20

Enter value for supplier_name: raji

old 1: insert into sales_ord_detail values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rate,&profit_percent,&sell_price,'&supplier_name')

new 1: insert into sales_ord_detail values(1235,1131,'lipstick',10,15,15,5,20,'raji')

1 row created.

SQL> /

Enter value for s_o_no: 1236

Enter value for product_no: 1141

Enter value for description: soap

Enter value for qty_ordered: 30

Enter value for qty_disp: 30

Enter value for product_rate: 22.5

Enter value for profit_percent: 10

Enter value for sell_price: 32.5

Enter value for supplier_name: kumar

old 1: insert into sales_ord_detail values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rate,&profit_percent,&sell_price,'&supplier_name')

new 1: insert into sales_ord_detail values(1236,1141,'soap',30,30,22.5,10,32.5,'kumar')

1 row created.

SQL> /

Enter value for s_o_no: 1237

Enter value for product_no: 1151

Enter value for description: kajal

Enter value for qty_ordered: 25

Enter value for qty_disp: 18

Enter value for product_rate: 92

Enter value for profit_percent: 7

Enter value for sell_price: 99

Enter value for supplier_name: devi

old 1: insert into sales_ord_detail values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rate,&profit_percent,&sell_price,'&supplier_name')

new 1: insert into sales_ord_detail values(1237,1151,'kajal',25,18,92,7,99,'devi')

1 row created.

SQL> /

Enter value for s_o_no: 1238

Enter value for product_no: 1161

Enter value for description: hairoil

Enter value for qty_ordered: 5

Enter value for qty_disp: 30

Enter value for product_rate: 10

Enter value for profit_percent: 5

Enter value for sell_price: 15

Enter value for supplier_name: hari

old 1: insert into sales_ord_detail values(&s_o_no,&product_no,'&description',&qty_ordered,&qty_disp,&product_rate,&profit_percent,&sell_price,'&supplier_name')

new 1: insert into sales_ord_detail values(1238,1161,'hairoil',5,30,10,5,15,'hari')

1 row created.

SQL> select *from sales_ord_detail;

S_O_NO	PRODUCT_NO	DESCRIPTION	QTY_ORDERED	QTY_DISP	PRODUCT_RATE	PROFIT_PERCENT	SELL_PRICE	SUPPLIER_N
1234	1121	shampoo	20	10	10	20.5	30.5	raj
1235	1131	lipstick	10	15	5	15	20	raji
1236	1141	soap	30	30	10	22.5	32.5	kumar
1237	1151	kajal	25	18	7	92	99	devi
1238	1161	hairoil	5	30	10	5	15	hari

a.Select each row and comput sell price *0.50.

SQL> select sell_price*0.50 from sales_ord_detail;

SELL_PRICE*0.50

15.25

10

16.25

49.5

7.5

Select each row and comput sell price *1.50.

SQL> select sell_price*1.50 from sales_ord_detail;

SELL_PRICE*1.50

45.75

30

48.75

148.5

22.5

b.Select product_no,profit_percent,sell_price from sales_ord_detail where profit_percent not between 10 and20 both inclusive.

SQL>select product_no,profit_percent,sell_price from sales_ord_detail where profit_percent not between 10 and 20;

PRODUCT_NO	PROFIT_PERCENT	SELL_PRICE
1131	5	20
1151	7	99
1161	5	15

SQL>select product_no,profit_percent,sell_price from sales_ord_detail where profit_percent= 10 and profit_percent=20;

PRODUCT_NO	PROFIT_PERCENT	SELL_PRICE
1121	10	30.5
1141	10	32.5

c.select product_no,description,profit_percent,sell_price from sales_ord_detail where profit_percent is not between 20 and 30;

SQL>select product_no,description,profit_percent,sell_price from sales_ord_detail where profit_percent not between 20 and 30;

PRODUCT_NO	DESCRIPTION	PROFIT_PERCENT	SELL_PRICE
1131	lipstick	5	20
1151	kajal	7	99
1161	hairoil	5	15

d.select supplier_name and product_no where supplier_name has ‘r’ or ’h’.

```
SQL> select supplier_name,product_no from sales_ord_detail where supplier_name like
        '_%r__';

no rows selected


SQL> select supplier_name,product_no from sales_ord_detail where supplier_name like
        '_%h__';

no rows selected
```

Ex. No: 5

Date:

CREATION OF DEPARTMENT AND EMPLOYEE DATABASE USING SQL QUERIES

Aim:

To create the Department and Employee tables with the required fields to process the instructions using SQL Queries.

QUERIES:

I. Table creation dept.

```
SQL> create table dept(dno number(5),
  2  dname varchar2(15),
  3  primary key(dno));

Table created.

SQL> desc dept;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DNO                                       NOT NULL NUMBER(5)
 DNAME                                              VARCHAR2(15)
SQL> insert into dept values(&dno,'&dname');
Enter value for dno: 10
Enter value for dname: accounting
old   1: insert into dept values(&dno,'&dname')
new   1: insert into dept values(10,'accounting')
1 row created.
```

```
SQL> /
Enter value for dno: 20
Enter value for dname: research
old   1: insert into dept values(&dno,'&dname')
new   1: insert into dept values(20,'research')
1 row created.

SQL> /
Enter value for dno: 30
Enter value for dname: sales
old   1: insert into dept values(&dno,'&dname')
new   1: insert into dept values(30,'sales')
1 row created.

SQL> /
Enter value for dno: 40
Enter value for dname: operations
old   1: insert into dept values(&dno,'&dname')
new   1: insert into dept values(40,'operations')
1 row created.
SQL> select *from dept;
     DNO            DNAME
     ----------     ---------------
      10            accounting
      20            research
      30            sales
      40            operations
```

II. Table creation employee.

```
SQL> create table employee(eno number(5),
  2  ename varchar2(15),
  3  job_type varchar2(15),
  4  hiredate date,
  5  dno number(5),
  6  salary number(7,2),
  7  primary key(eno),
  8  foreign key(dno) references dept(dno));
Table created.

SQL> desc employee;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ENO                                       NOT NULL NUMBER(5)
 ENAME                                              VARCHAR2(15)
 JOB_TYPE                                           VARCHAR2(15)
 HIREDATE                                           DATE
 DNO                                                NUMBER(5)
 SALARY                                             NUMBER(7,2)

SQL> insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary);
Enter value for eno: 7369
Enter value for ename: smith
Enter value for job_type: clerk
Enter value for hiredate: 17-dec-80
Enter value for dno: 20
```

Enter value for salary: 800

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7369,'smith','clerk','17-dec-80',20,800)

1 row created.

SQL> /

Enter value for eno: 7499

Enter value for ename: allen

Enter value for job_type: salesman

Enter value for hiredate: 20-feb-81

Enter value for dno: 20

Enter value for salary: 1600

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7499,'allen','salesman','20-feb-81',20,1600)

1 row created.

SQL> /

Enter value for eno: 7521

Enter value for ename: ward

Enter value for job_type: manager

Enter value for hiredate: 22-feb-81

Enter value for dno: 10

Enter value for salary: 1250

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7521,'ward','manager','22-feb-81',10,1250)

1 row created.

SQL> /

Enter value for eno: 7566

Enter value for ename: jones

Enter value for job_type: salesman

Enter value for hiredate: 02-apr-81

Enter value for dno: 30

Enter value for salary: 2975

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7566,'jones','salesman','02-apr-81',30,2975)

1 row created.

SQL> /

Enter value for eno: 7654

Enter value for ename: martin

Enter value for job_type: manager

Enter value for hiredate: 28-sep-81

Enter value for dno: 20

Enter value for salary: 1250

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7654,'martin','manager','28-sep-81',20,1250)

1 row created.

SQL> /

Enter value for eno: 7698

Enter value for ename: blake

Enter value for job_type: manager

Enter value for hiredate: 01-may-81

Enter value for dno: 30

Enter value for salary: 2850

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7698,'blake','manager','01-may-81',30,2850)

1 row created.

SQL> /

Enter value for eno: 7782

Enter value for ename: clark

Enter value for job_type: analyst

Enter value for hiredate: 09-jun-81

Enter value for dno: 20

Enter value for salary: 2450

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7782,'clark','analyst','09-jun-81',20,2450)

1 row created.

SQL> /

Enter value for eno: 7788

Enter value for ename: scott

Enter value for job_type: salesman

Enter value for hiredate: 19-apr-87

Enter value for dno: 30

Enter value for salary: 3000

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7788,'scott','salesman','19-apr-87',30,3000)

1 row created.

SQL> /

Enter value for eno: 7839

Enter value for ename: king

Enter value for job_type: president

Enter value for hiredate: 17-nov-81

Enter value for dno: 20

Enter value for salary: 5000

```
old   1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)
new   1: insert into employee values(7839,'king','president','17-nov-81',20,5000)
1 row created.
SQL> /
Enter value for eno: 7844
Enter value for ename: turner
Enter value for job_type: clerk
Enter value for hiredate: 08-sep-81
Enter value for dno: 30
Enter value for salary: 1500
old   1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)
new   1: insert into employee values(7844,'turner','clerk','08-sep-81',30,1500)
1 row created.
SQL> /
Enter value for eno: 7876
Enter value for ename: adams
Enter value for job_type: clerk
Enter value for hiredate: 23-may-87
Enter value for dno: 20
Enter value for salary: 1100
old   1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)
new   1: insert into employee values(7876,'adams','clerk','23-may-87',20,1100)
1 row created.
SQL> /
Enter value for eno: 7900
Enter value for ename: james
Enter value for job_type: salesman
```

Enter value for hiredate: 03-dec-81

Enter value for dno: 10

Enter value for salary: 900

old 1: insert into employee values(&eno,'&ename','&job_type','&hiredate',&dno,&salary)

new 1: insert into employee values(7900,'james','salesman','03-dec-81',10,900)

1 row created.

SQL> select *from employee;

ENO	ENAME	JOB_TYPE	HIREDATE	DNO	SALARY
7369	smith	clerk	17-DEC-80	20	800
7499	allen	salesman	20-FEB-81	20	1600
7521	ward	manager	22-FEB-81	10	1250
7566	jones	salesman	02-APR-81	30	2975
7654	martin	manager	28-SEP-81	20	1250
7698	blake	manager	01-MAY-81	30	2850
7782	clark	analyst	09-JUN-81	20	2450
7788	scott	salesman	19-APR-87	30	3000
7839	king	president	17-NOV-81	20	5000
7844	turner	clerk	08-SEP-81	30	1500
7876	adams	clerk	23-MAY-87	20	1100
7900	james	salesman	03-DEC-81	10	900

12 rows selected.

SQL> alter table employee add(manager number(5),commission number(10,2));

Table altered.

SQL> alter table employee add foreign key(manager)references employee(eno);

Table altered.

SQL> desc employee;

Name	Null?	Type
ENO	NOT NULL	NUMBER(5)
ENAME		VARCHAR2(15)
JOB_TYPE		VARCHAR2(15)
HIREDATE		DATE
DNO		NUMBER(5)
SALARY		NUMBER(7,2)
MANAGER		NUMBER(5)
COMMISSION		NUMBER(10,2)

SQL> update employee set commission=300 where ename='allen';

1 row updated.

SQL> update employee set commission=500 where ename='ward';

1 row updated.

SQL> update employee set commission=1400 where ename='martin';

1 row updated.

SQL> update employee set commission=0 where ename='king';

1 row updated.

SQL> select *from employee;

ENO	ENAME	JOB_TYPE	HIREDATE	DNO	SALARY	MANAGER	COMMISSION
7369	smith	clerk	17-DEC-80	20	800		
7499	allen	salesman	20-FEB-81	20	1600		300
7521	ward	manager	22-FEB-81	10	1250		500
7566	jones	salesman	02-APR-81	30	2975		
7654	martin	manager	28-SEP-81	20	1250		1400
7698	blake	manager	01-MAY-81	30	2850		
7782	clark	analyst	09-JUN-81	20	2450		
7788	scott	salesman	19-APR-87	30	3000		
7839	king	president	17-NOV-81	20	5000		0
7844	turner	clerk	08-SEP-81	30	1500		
7876	adams	clerk	23-MAY-87	20	1100		
7900	james	salesman	03-DEC-81	10	900		

12 rows selected.

SQL> update employee set manager=7499 where ename='smith';

1 row updated.

SQL> update employee set manager=7698 where ename='allen';

1 row updated.

SQL> update employee set manager=7698 where ename='ward';

1 row updated.

SQL> update employee set manager=7839 where ename='jones';

1 row updated.

SQL> update employee set manager=7698 where ename='martin';

1 row updated.

SQL> update employee set manager=7839 where ename='blake';

1 row updated.

SQL> update employee set manager=7839 where ename='clark';

1 row updated.

SQL> update employee set manager=7566 where ename='scott';

1 row updated.

SQL> update employee set manager=7698 where ename='turner';

1 row updated.

SQL> update employee set manager=7788 where ename='adams';

1 row updated.

SQL> update employee set manager=7698 where ename='james';

1 row updated.

SQL> select *from employee;

ENO	ENAME	JOB_TYPE	HIREDATE	DNO	SALARY	MANAGER	COMMISSION
7369	smith	clerk	17-DEC-80	20	800	7369	
7499	allen	salesman	20-FEB-81	20	1600	7698	300
7521	ward	manager	22-FEB-81	10	1250	7698	500
7566	jones	salesman	02-APR-81	30	2975	7839	
7654	martin	manager	28-SEP-81	20	1250	7698	1400
7698	blake	manager	01-MAY-81	30	2850	7839	
7782	clark	analyst	09-JUN-81	20	2450	7839	
7788	scott	salesman	19-APR-87	30	3000	7566	
7839	king	president	17-NOV-81	20	5000		0
7844	turner	clerk	08-SEP-81	30	1500	7698	
7876	adams	clerk	23-MAY-87	20	1100	7788	
7900	james	salesman	03-DEC-81	10	900	7698	

12 rows selected.

SQL> select eno,job_type,hiredate,ename from employee order by eno;

ENO	JOB_TYPE	HIREDATE	ENAME
7369	clerk	17-DEC-80	smith
7499	salesman	20-FEB-81	allen
7521	manager	22-FEB-81	ward
7566	salesman	02-APR-81	jones
7654	manager	28-SEP-81	martin
7698	manager	01-MAY-81	blake
7782	analyst	09-JUN-81	clark

7788	salesman	19-APR-87	scott
7839	president	17-NOV-81	king
7844	clerk	08-SEP-81	turner
7876	clerk	23-MAY-87	adams
7900	salesman	03-DEC-81	james

12 rows selected.

```
SQL> select distinct job_type from employee;

JOB_TYPE
---------------
salesman
clerk
president
manager
analyst

SQL> select ename||','||job_type from employee;

ENAME||','||JOB_TYPE
------------------------------
smith,clerk
allen,salesman
ward,manager
jones,salesman
martin,manager
blake,manager
clark,analyst
scott,salesman
```

king,president

turner,clerk

adams,clerk

ENAME||','||JOB_TYPE

---

james,salesman

12 rows selected.

SQL> select ename,salary from employee where salary>2850;

ENAME SALARY

--- ---

jones 2975

scott 3000

king 5000

SQL> select ename,dno from employee where eno=7900;

ENAME DNO

--- ---

james 10

SQL> select *from employee where salary<>1500 and salary>2850;

ENO	ENAME	JOB_TYPE	HIREDATE	DNO	SALARY	MANAGER	COMMISSION--
7566	jones	salesman	02-APR-81	30	2975	7839	
7788	scott	salesman	19-APR-87	30	3000	7566	
7839	king	president	17-NOV-81	20	5000		0

SQL> select ename,dno from employee where dno=20 and dno=30 order by ename;

no rows selected

SQL> select ename,dno from employee where dno=20 or dno=30 order by ename;

ENAME	DNO
adams	20
allen	20
blake	30
clark	20
jones	30
king	20
martin	20
scott	30
smith	20
turner	30

10 rows selected.

SQL> select *from employee where job_type<>'manager';

ENO	ENAME	JOB_TYPE	HIREDATE	DNO	SALARY	MANAGER	COMMISSION
7369	smith	clerk	17-DEC-80	20	800	7499	
7499	allen	salesman	20-FEB-81	20	1600	7698	300
7566	jones	salesman	02-APR-81	30	2975	7839	
7782	clark	analyst	09-JUN-81	20	2450	7839	
7788	scott	salesman	19-APR-87	30	3000	7566	

7839	king	president	17-NOV-81	20	5000	0
7844	turner	clerk	08-SEP-81	30	1500	7698
7876	adams	clerk	23-MAY-87	20	1100	7788
7900	james	salesman	03-DEC-81	10	900	7698

9 rows selected.

SQL> select ename,salary,commission from employee where commission>=0;

ENAME	SALARY	COMMISSION
allen	1600	300
ward	1250	500
martin	1250	1400
king	5000	0

SQL> select ename,commission,salary from employee order by salary desc,commission desc;

ENAME	COMMISSION	SALARY
king	0	5000
scott		3000
jones		2975
blake		2850
clark		2450
allen	300	1600

turner 1500

martin 1400 1250

ward 500 1250

adams 1100

james 900

smith 800

12 rows selected.

SQL> select ename from employee where ename like '%R' or dno=30 or manager=7788;

ENAME

---

jones

blake

scott

turner

adams

SQL> select ename,commission,salary+(salary*5/100) from employee where commission>14;

ENAME	COMMISSION	SALARY+(SALARY*5/100)
allen	300	1680
ward	500	1312.5
martin	1400	1312.5

```
SQL> select ename from employee where ename like '%__a%';

ENAME
----------
blake
clark
adams

SQL> select initcap(ename) from employee where ename like 'G%';

no rows selected

SQL> select initcap(ename) from employee where ename like 'a%';

INITCAP(ENAME)
----------
Allen
Adams

SQL> select initcap(ename) from employee where ename like 'm%';

INITCAP(ENAME)
----------
Martin

SQL> select employee.ename,dept.dno,dname from employee,dept where
     dept.dno=employee.dno;

ENAME          DNO DNAME
---------- ---------- ----------
smith              20          research
```

allen 20 research

ward 10 accounting

jones 30 sales

martin 20 research

blake 30 sales

clark 20 research

scott 30 sales

king 20 research

turner 30 sales

adams 20 research

james 10 accounting

12 rows selected.

SQL> select job_type from employee where dno=30;

JOB_TYPE

---

salesman

manager

salesman

clerk

SQL> alter table dept add(location varchar2(20));

Table altered.

SQL> select*from dept;

DNO DNAME LOCATION

--- --- ---

10 accounting

20 research

30 sales

40 operations

SQL> update dept set location='mumbai' where dno=10;

1 row updated.

SQL> update dept set location='newyork' where dno=20;

1 row updated.

SQL> update dept set location='dallas' where dno=30;

1 row updated.

SQL> update dept set location='chihago' where dno=40;

1 row updated.

SQL> select *from dept;

DNO	DNAME	LOCATION
10	accounting	mumbai
20	research	newyork
30	sales	dallas
40	operations	chihago

SQL> select employee.ename, employee.job_type, dept.dno,dept.dname from employee,dept where dept.location='mumbai'and dept.dno=employee.dno;

ENAME JOB_TYPE DNO DNAME

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

ward manager 10 accounting

james salesman 10 accounting

SQL> select ename,dno,salary from employee where commission<>0;

ENAME DNO SALARY

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

allen 20 1600

ward 10 1250

martin 20 1250

SQL> select max(salary)from employee;

MAX(SALARY)

5000

SQL> select min(salary)from employee;

MIN(SALARY)

800

SQL> select avg(salary)from employee;

AVG(SALARY)

2056.25

```
SQL> select sum(salary)from employee;

SUM(SALARY)
-----------
      24675

SQL> select max(salary),min(salary),avg(salary),sum(salary)from employee;

MAX(SALARY)    MIN(SALARY)        AVG(SALARY)        SUM(SALARY)
-----------    -----------        -----------        -----------
      5000            800            2056.25              24675
SQL> select employee.eno,ename from employee where salary>2056.25;
     ENO    ENAME
----------    ---------------
      7566    jones
      7698    blake
      7782    clark
      7788    scott
      7839    king
SQL>  select employee.eno,ename,dept.dname from employee,dept where ename
      like'%t%' and employee.dno=dept.dno;
     ENO    ENAME         DNAME
---------- --------------- ---------------
      7369     smith          research
      7654    martin          research
      7788    scott             sales
      7844    turner             sales
```

Ex. No: 6

Date:

FACTORIAL VALUE CALCULATION

Aim:

To calculate the factorial value using PL/SQL

Program:

```
Declare
    i number;
    f number;
    n number;
Begin
    n:=&n;
    f:=1;
    for i  in 1 ..n loop
    f:=f*i;
    end loop;
dbms_output.put_line('factorial value: ' || f);
end;
/
```

OUTPUT:

SQL> ed fact.sql

SQL> set serveroutput on

SQL> @fact.sql

Enter value for n: 2

old 6: n:=&n;

new 6: n:=2;

factorial value: 2

PL/SQL procedure successfully completed

Ex. No: 7

Date:

COUNT THE TOTAL NUMBER OF EMPLOYEES - USING FUNCTION

Aim:

To count the total number of employees using function in PL/SQL

Program:

```
SQL>create table emp021 (ename varchar2(20),empno number(5),dno
    number(5),dname varchar2(20));

Table created.

SQL>Insert into emp021 values('&ename',&empno,&dno,'&dname');

Enter value for ename:monika

Enter value for empno:1001

Enter value for dno:10

Enter value for dname:cs

1 row created

SQL>/

SQL> select*from emp021;

ENAME          EMPNO     DNO   DNAME

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

monika          1001        10      cs

vithya          1002        20      bcom
```

yuva 1003 10 maths

viji 1004 20 cs

```
create or replace function deptcount(a emp021.dno%type)
return number is
total number(2):=0;
begin
    select count(*) into total from emp021 where dno=a;
    return total;
    end;
    /
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @funct1.sql
Function created.

SQL>ed funct2.sql

Declare
    cnt number(2);
    dn emp021.dno%type;
Begin
   dn:=&dn;
   cnt:=deptcount(dn);
```

```
dbms_output.put_line('department number:'|| dn);
dbms_output.put_line('total employee with this code:'|| cnt);
end;
/
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @funct2.sql
Enter value for dn: 10
old   5:     dn:=&dn;
new   5:     dn:=10;
department number:10
total employee with this code:2

PL/SQL procedure successfully completed.
```

Ex. No: 8

Date:

UPDATE THE FIELDS OF THE TABLE - USING IMPLICIT CURSOR

Aim:

To update the fields of the student table using implicit cursor in PL/SQL

Program:

```
SQL> create table stu21 (name varchar2(20),rollno number(7),mark1 number(3),mark2
     number(3));

Table created.

SQL> insert into  stu21 values('&name',&rollno,&mark1,&mark2);

Enter value for name: monika

Enter value for rollno: 1111

Enter value for mark1:78

Enter value for mark2: 80

old   1: insert into  student values('&name',&rollno,&mark1,&mark2)

new   1: insert into  student values('monika',1111,78,80)

1 row created.

SQL>select*from stu21;

NAME        ROLLNO     MARK1     MARK2     MARK3

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

monika        1111          78        80          60

vithya        1112          88        90          87

yuva          1112          76        58          77

brindha       1115          56        76          77

paru          1116          44        77          71
```

```
SQL> /
SQL> ed impcur.sal


Declare
     no number;
     m1 number;
     m2 number;
     m3 number;
Begin
    no:=&no;
    m1:=&m1;
    m2:=&m2;
    m3:=&m3;
update stu21 set mark1=m1,mark2=m2,mark3=m3 where rollno=no;
if sql% found then
dbms_output.put_line('record updated');
else
dbms_output.put_line('record not updated');
end if;
end;
/
```

OUTPUT: 1

```
SQL> set serveroutput on
SQL> @impcursor.sql
Enter value for no: 1111
old   7:     no:=&no;
new   7:     no:=1111;
Enter value for m1: 100
old   8:     m1:=&m1;
new   8:     m1:=100;
Enter value for m2: 100
old   9:     m2:=&m2;
new   9:     m2:=100;
Enter value for m3: 100
old  10:     m3:=&m3;
new  10:     m3:=100;
```

record updated

PL/SQL procedure successfully completed.

```
SQL> select*from stu21;
```

NAME	ROLLNO	MARK1	MARK2	MARK3
monika	1111	100	100	100
vithya	1112	88	90	87
yuva	1112	76	58	77
brindha	1115	56	76	77
paru	1116	44	77	71

OUTPUT : 2

```
SQL> @impcursor.sql
Enter value for no: 1000
old   7:     no:=&no;
new   7:     no:=1000;
Enter value for m1: 12
old   8:     m1:=&m1;
new   8:     m1:=12;
Enter value for m2: 12
old   9:     m2:=&m2;
new   9:     m2:=12;
Enter value for m3: 12
old  10:     m3:=&m3;
new  10:     m3:=12;
```

record not updated

PL/SQL procedure successfully completed.

SQL>select*from stu21;

NAME	ROLLNO	MARK1	MARK2	MARK3
monika	1111	78	80	60
vithya	1112	88	90	87
yuva	1112	76	58	77
brindha	1115	56	76	77
paru	1116	44	77	71

Ex. No: 9

Date:

UPDATE THE EMPLOYEE'S COMMISSION DETAILS - USING PROCEDURE

Aim:

To update the employee's commission details using procedure in PL/SQL

Program:

```
SQL>create table empp21(empno number(5),depcode number(5),salary
    number(10),commission number(7));

Table created.


SQL>insert into empp21 values(&empno,&depcode,&salary,&commission);

Enter value for empno: 111

Enter value for depcode: 101

Enter value for salary: 10000

Enter value for commission: 0


SQL> select*from empp21;


    EMPNO    DEPCODE     SALARY COMMISSION

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

     111            101            10000       0

     112            102            15000       0

     113            103            16000       0

     114            104            14000       0
```

```
Create or replace procedure increases(s in empp21.depcode%type,p number)
as
Begin
  update empp21 set commission=salary*(p/100)
  where depcode=s;
  end;
  /
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @proc.sql
Procedure created.
SQL>exec increases(101,20);
PL/SQL procedure successfully completed.

SQL>exec increases(102,10);
PL/SQL procedure successfully completed.
SQL>select*from empp21;
```

EMPNO	DEPCODE	SALARY	COMMISSION
111	101	10000	2000
112	101	15000	3000
113	102	16000	1600
114	104	14000	0

Ex. No: 10

Date:

DISPLAY THE DETAILS OF EMPLOYEES - USING EXPLICIT CURSOR

Aim:

To display the details of the employees using explicit cursor in PL/SQL

Program:

```
SQL> create table empl(empno number(5),empname varchar2(20),empjob
     varchar2(20),jobcode number(5),salary number(5));

Table created.


SQL>insert into empl values(&empno,'&empname','&empjob',&jobcode,&salary)

Enter value for empno: 112

Enter value for empname: yuva

Enter value for empjob: programmer

Enter value for jobcode: 12

Enter value for salary: 2000

old   1: insert into empl values(&empno,'&empname','&empjob',&jobcode,&salary)

new   1: insert into empl values(112,'yuva','programmer',12,2000)

1 row created.

SQL> /
```

SQL>select*from empl;

EMPNO	EMPNAME	EMPJOB	JOBCODE	SALARY
111	monika	clerk	12	1500
112	yuva	programmer	12	2000
113	vithya	salesman	12	3000
114	viji	manager	11	2500

SQL> ed excursor.sql

```
declare
    eno empl.empno % type;
    ename empl.empname % type;
    ejob   empl.empjob % type;
    jcode empl.jobcode% type;
    esal  empl.salary% type;
cursor C is select*from empl where jobcode=11;
begin
   OPEN C;
loop
  FETCH C into eno,ename,ejob,jcode,esal;
exit when C %  notfound;
dbms_output.put_line('emp number: ' || eno);
dbms_output.put_line('emp name: ' || ename);
dbms_output.put_line('emp job: ' || ejob);
```

```
dbms_output.put_line('jobcode: ' || jcode);
dbms_output.put_line(' salary: ' || esal);
dbms_output.put_line('_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ ');
end loop;
CLOSE C;
end;
/
```

OUTPUT:

```
SQL>set serveroutput on
SQL>@excursor.sql

Emp number : 112
Emp name    : yuva
Emp job       : programmer
Job code       : 12
Emp salary    : 2000
-----------------------------------
Emp number : 111
Emp name    : monika
Emp job       : clerk
Job code       : 12
Emp salary    : 1500
-----------------------------------
Emp number : 113
Emp name    : vithya
```

Emp job : salesman

Job code : 12

Emp salary : 3000

PL/SQL procedure successfully completed.

Emp number : 112

Emp name : yuva

Emp job : programmer

Job code : 12

Emp salary : 2000

PL/SQL procedure successfully completed

Ex. No: 11

Date:

PERFORM ARITHMETIC OPERATIONS USING PACKAGE

Aim:

To perform the arithmetic operations using package in PL/SQL

Program:

```
SQL> ed pack1.sql

Create or replace package pk as
function add(x number,y number)return number;
function sub(x number,y number)return number;
function mul(x number,y number)return number;
function div(x number,y number)return number;
end;
/
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @pack1.sql
Package created.
```

```
SQL> ed pack2.sql;


Create or replace package body pk as
function add(x number, y number) return number is
begin
        return(x+y);
        end add;
function sub(x number, y number) return number is
begin
       return(x-y);
       end sub;
function mul(x number, y number) return number  is
begin
      return(x*y);
      end mul;
function div(x number, y number) return number is
begin
      return(x/y);
      end div;
      end pk;
       /
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @pack2.sql;
Package body created.
```

SQL> ed pack3.sql;

```
Declare
        a number;
        b number;
        n1 number;
        n2 number;
        n3 number;
        n4 number;
Begin
        a:= &a;
        b:= &b;
        n1:= pk.add(a,b);
        n2:= pk.sub(a,b);
        n3:= pk.mul(a,b);
        n4:= pk.div(a,b);
dbms_output.put_line('Addition: ' || n1);
dbms_output.put_line('Substraction: ' || n2);
dbms_output.put_line('Multiplication: ' || n3);
dbms_output.put_line('Division: ' || n4);
end;
/
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @pack3.sql;
```

```
Enter value for a: 3
old   9:        a:= &a;
new   9:        a:= 3;
Enter value for b: 3
old  10:        b:= &b;
new  10:        b:= 3;
Addition: 6
Substraction: 0
Multiplication: 9
Division: 1

PL/SQL procedure successfully completed.
```

Ex. No: 12

Date:

UPDATE THE EMPLOYEE'S SALARY DETAILS - USING TRIGGER

Aim:

To update employee's salary details using trigger in PL/SQL

Program:

```
SQL> ed tri.sql;

Create or replace trigger mysal before insert or
update of sal on employeee for each row
Begin
if:NEW.sal>10000 then
RAISE_APPLICATION_ERROR(-20000,'incorrect value');
end if;
end;
/

SQL> create table employeee(id number(5),sal number(6));
Table created.
```

OUTPUT:

```
SQL> set serveroutput on
SQL> @tri.sql;
Trigger created.
```

```
SQL> insert into employeee values(&id,&sal);
Enter value for id: 111
Enter value for sal: 15000
old   1: insert into employeee values(&id,&sal)
new   1: insert into employeee values(111,15000)
insert into employeee values(111,15000)
            *
ERROR at line 1:
ORA-20000: incorrect value
ORA-06512: at "SYSTEM.MYSAL", line 3
ORA-04088: error during execution of trigger 'SYSTEM.MYSAL'


SQL> /
Enter value for id: 111
Enter value for sal: 8000
old   1: insert into employeee values(&id,&sal)
new   1: insert into employeee values(111,8000)
1 row created.
SQL> /
Enter value for id: 112
Enter value for sal: 9000
old   1: insert into employeee values(&id,&sal)
new   1: insert into employeee values(112,9000)
```

1 row created.

SQL> update employeee set sal= 11000 where id=112;

update employeee set sal= 11000 where id=112

*

ERROR at line 1:

ORA-20000: incorrect value

ORA-06512: at "SYSTEM.MYSAL", line 3

ORA-04088: error during execution of trigger 'SYSTEM.MYSAL'

SQL> update employeee set sal= 7500 where id=112;

1 row updated.

SQL> select*from employeee;

ID	SAL
111	8000
112	7500