HANDS ON EXERCISES

1. **Create the tables described below:**

Table Name: CLIENT_MASTER
Description: Used to store client information.

Column Name	Data Type	Size	Default	Attributes
CLIENTNO	Varchar2	6		
NAME	Varchar2	20		
ADDRESS1	Varchar2	30		
ADDRESS2	Varchar2	30		
CITY	Varchar2	15		
PINCODE	Number	8		
STATE	Varchar2	15		
BALDUE	Number	10,2		

Table Name: PRODUCT_MASTER
Description: Used to store product information.

Column Name	Data Type	Size	Default	Attributes
PRODUCTNO	Varchar2	6		
DESCRIPTION	Varchar2	15		
PROFITPERCENT	Number	4,2		
UNITMEASURE	Varchar2	10		
QTYONHAND	Number	8		
REORDERLVL	Number	8		
SELLPRICE	Number	8,2		
COSTPRICE	Number	8,2		

Table Name: SALESMAN_MASTER
Description: Used to store salesman information working for the company.

Column Name	Data Type	Size	Default	Attributes
SALESMANNO	Varchar2	6		
SALESMANNAME	Varchar2	20		
ADDRESS1	Varchar2	30		
ADDRESS2	Varchar2	30		
CITY	Varchar2	20		
PINCODE	Number	8		
STATE	Varchar2	20		
SALAMT	Number	8,2		
TGTTOGET	Number	6,2		
YTDSALES	Number	6,2		
REMARKS	Varchar2	60		

2. Insert the following data into their respective tables:

a) Data for **CLIENT_MASTER** table:

ClientNo	Name	City	Pincode	State	BalDue
C00001	Ivan Bayross	Mumbai	400054	Maharashtra	15000
C00002	Mamta Muzumdar	Madras	780001	Tamil Nadu	0
C00003	Chhaya Bankar	Mumbai	400057	Maharashtra	5000
C00004	Ashwini Joshi	Bangalore	560001	Karnataka	0
C00005	Hansel Colaco	Mumbai	400060	Maharashtra	2000
C00006	Deepak Sharma	Mangalore	560050	Karnataka	0

b) Data for **PRODUCT_MASTER** table:

ProductNo	Description	Profit Percent	Unit Measure	QtyOn Hand	ReorderLvl	SellPrice	CostPrice
P00001	T-Shirts	5	Piece	200	50	350	250
P0345	Shirts	6	Piece	150	50	500	350
P06734	Cotton Jeans	5	Piece	100	20	600	450
P07865	Jeans	5	Piece	100	20	750	500
P07868	Trousers	2	Piece	150	50	850	550
P07885	Pull Overs	2.5	Piece	80	30	700	450
P07965	Denim Shirts	4	Piece	100	40	350	250
P07975	Lycra Tops	5	Piece	70	30	300	175
P08865	Skirts	5	Piece	75	30	450	300

c) Data for **SALESMAN_MASTER** table:

SalesmanNo	Name	Address1	Address2	City	PinCode	State
S00001	Aman	A/14	Worli	Mumbai	400002	Maharashtra
S00002	Omkar	65	Nariman	Mumbai	400001	Maharashtra
S00003	Raj	P-7	Bandra	Mumbai	400032	Maharashtra
S00004	Ashish	A/5	Juhu	Mumbai	400044	Maharashtra

SalesmanNo	SalAmt	TgtToGet	YtdSales	Remarks
S00001	3000	100	50	Good
S00002	3000	200	100	Good
S00003	3000	200	100	Good
S00004	3500	200	150	Good

3. Exercise on retrieving records from a table

a. Find out the names of all the clients.
b. Retrieve the entire contents of the Client_Master table.
c. Retrieve the list of names, city and the sate of all the clients.
d. List the various products available from the Product_Master table.
e. List all the clients who are located in Mumbai.
f. Find the names of salesmen who have a salary equal to Rs.3000.

4. Exercise on updating records in a table

a. Change the city of ClientNo 'C00005' to 'Bangalore'.
b. Change the BalDue of ClientNo 'C00001' to Rs. 1000.
c. Change the cost price of 'Trousers' to Rs. 950.00.
d. Change the city of the salesman to Pune.

5. Exercise on deleting records in a table
a. Delete all salesmen from the Salesman_Master whose salaries are equal to Rs. 3500.
b. Delete all products from Product_Master where the quantity on hand is equal to 100.
c. Delete from Client_Master where the column state holds the value 'Tamil Nadu'.

6. Exercise on altering the table structure
a. Add a column called 'Telephone' of data type 'number' and size ='10' to the Client_Master table.
b. Change the size of SellPrice column in Product_Master to 10,2.

7. Exercise on deleting the table structure along with the data
a. Destroy the table Client_Master along with its data.

8. Exercise on renaming the table
a. Change the name of the Salesman_Master table to sman_mast.

10. The default behavior of the foreign key can be changed by using the _____ option.
11. _____ constraints must be specified as a logical expression that evaluates either to TRUE or FALSE.
12. In a CHECK constraint the condition must be a _____ expression that can be evaluated using the values in the row being inserted or updated.
13. _____ constraints can be defined using the constraint clause, in the ALTER TABLE command.
14. Dropping UNIQUE and PRIMARY KEY constraints also drops all associated _____.

TRUE OR FALSE

15. Business rules that have to be applied to data are completely System dependent.
16. Constraints super control the data being entered into a table for temporary storage.
17. A NULL value is equivalent to a value of zero.
18. Setting a NULL value is appropriate when the actual value is unknown.
19. A table cannot contain multiple unique keys.
20. Oracle ignores any UNIQUE, FOREIGN KEY, CHECK constraints on a NULL value.
21. A primary key column in a table is an optional column.
22. Standard business rules do not allow multiple entries for the same product.
23. The master table can be referenced in the foreign key definition by using the clause REFERENCES tablename.columname when defining the foreign key.
24. A CHECK constraint consists of subqueries and sequences.
25. The USER_CONSTRAINTS command displays only the column names, data type, size and the NOT NULL constraint.
26. Drop the constraint using the DROP TABLE command with the DELETE clause.
27. At the time of table creation a default value can be assigned to a column.
28. If a column level constraint is defined on the column with a default value, the default value clause must precede the constraint definition.

HANDS ON EXERCISES

1. Create the tables described below:

Table Name: **CLIENT_MASTER**
Description: Used to store client information.

Column Name	Data Type	Size	Default	Attributes
CLIENTNO	Varchar2	6		Primary Key / first letter must start with 'C'
NAME	Varchar2	20		Not Null
ADDRESS1	Varchar2	30		

Details for **CLIENT_MASTER** table continued.

Column Name	Data Type	Size	Default	Attributes
ADDRESS2	Varchar2	30		
CITY	Varchar2	15		
PINCODE	Number	8		
STATE	Varchar2	15		
BALDUE	Number	10,2		

Table Name: **PRODUCT_MASTER**
Description: Used to store product information.

Column Name	Data Type	Size	Default	Attributes
PRODUCTNO	Varchar2	6		Primary Key / first letter must start with 'P'
DESCRIPTION	Varchar2	15		Not Null
PROFITPERCENT	Number	4,2		Not Null
UNITMEASURE	Varchar2	10		Not Null
QTYONHAND	Number	8		Not Null
REORDERLVL	Number	8		Not Null
SELLPRICE	Number	8,2		Not Null, Cannot be 0
COSTPRICE	Number	8,2		Not Null, Cannot be 0

Table Name: **SALESMAN_MASTER**
Description: Used to store salesman information working for the company.

Column Name	Data Type	Size	Default	Attributes
SALESMANNO	Varchar2	6		Primary Key / first letter must start with 'S'
SALESMANNAME	Varchar2	20		Not Null
ADDRESS1	Varchar2	30		Not Null
ADDRESS2	Varchar2	30		
CITY	Varchar2	20		
PINCODE	Number	8		
STATE	Varchar2	20		
SALAMT	Number	8,2		Not Null, Cannot be 0
TGTTOGET	Number	6,2		Not Null, Cannot be 0
YTDSALES	Number	6,2		Not Null
REMARKS	Varchar2	60		

Table Name: **SALES_ORDER**
Description: Used to store client's orders.

Column Name	Data Type	Size	Default	Attributes
ORDERNO	Varchar2	6		Primary Key / first letter must start with 'O'
CLIENTNO	Varchar2	6		Foreign Key references ClientNo of Client_Master table
ORDERDATE	Date			Not Null
DELYADDR	Varchar2	25		
SALESMANNO	Varchar2	6		Foreign Key references SalesmanNo of Salesman_Master table
DELYTYPE	Char	1	F	Delivery: part (P) / full (F)
BILLYN	Char	1		
DELYDATE	Date			Cannot be less than Order_Date
ORDERSTATUS	Varchar2	10		Values ('In Process', 'Fulfilled', 'BackOrder', 'Cancelled')

Table Name: SALES_ORDER_DETAILS
Description: Used to store client's orders with details of each product ordered.

Column Name	Data Type	Size	Default	Attributes
ORDERNO	Varchar2	6		Foreign Key references OrderNo of Sales_Order table
PRODUCTNO	Varchar2	6		Foreign Key references ProductNo of Product_Master table
QTYORDERED	Number	8		
QTYDISP	Number	8		
PRODUCTRATE	Number	10,2		

2. Insert the following data into their respective tables:

a) Re-insert the data generated for tables CLIENT_MASTER, PRODUCT_MASTER, and SALESMAN_MASTER. Refer to hands-on exercised for **Chapter 07:Interactive SQL-Part I.**

b) Data for Sales_Order table:

OrderNo	ClientNo	OrderDate	SalesmanNo	DelyType	BillYN	DelyDate	OrderStatus
O19001	C00001	12-June-04	S00001	F	N	20-July-02	In Process
O19002	C00002	25-June-04	S00002	P	N	27-June-02	Cancelled
O46865	C00003	18-Feb-04	S00003	F	Y	20-Feb-02	Fulfilled
O19003	C00001	03-Apr-04	S00001	F	Y	07-Apr-02	Fulfilled
O46866	C00004	20-May-04	S00002	P	N	22-May-02	Cancelled
O19008	C00005	24-May-04	S00004	F	N	26-July-02	In Process

c) Data for Sales_Order_Details table:

OrderNo	ProductNo	QtyOrdered	QtyDisp	ProductRate
O19001	P00001	4	4	525
O19001	P07965	2	1	8400
O19001	P07885	2	1	5250
O19002	P00001	10	0	525
O46865	P07868	3	3	3150
O46865	P07885	3	1	5250
O46865	P00001	10	10	525
O46865	P0345	4	4	1050
O19003	P03453	2	2	1050
O19003	P06734	1	1	12000
O46866	P07965	1	0	8400
O46866	P07975	1	0	1050
O19008	P00001	10	5	525
O19008	P07975	5	3	1050

22. The LENGTH function returns the length of a word.
23. The LTRIM returns char, with final characters removed after the last character not in the set. 'set' is optional, it defaults to spaces.
24. LPAD returns the string passed as a parameter after left padding it to a specified length.
25. The TO_CHAR (date conversion) converts a value of a NUMBER datatype to a character datatype, using the optional format string.
26. The DATE data type is used to store date and time information.
27. The TO_DATE() function also disallows part insertion of a DATE value into a column.
28. The ADD_MONTHS function returns date after adding the number of months specified in the function.
29. The TO-DATE function allows a user to insert date into a date column in any required format, by specifying the character value of the date to be inserted and its format.

HANDS ON EXERCISES

Using the tables created previously generate the SQL statements for the operations mentioned below. The tables in user are as follows:

a. Client_Master
b. Product_Master
c. Salesman_Master
d. Sales_Order
e. Sales_Order_Details

1. Perform the following computations on table data:

a. List the names of all clients having 'a' as the second letter in their names.
b. List the clients who stay in a city whose First letter is 'M'.
c. List all clients who stay in 'Bangalore' or 'Mangalore'
d. List all clients whose BalDue is greater than value 10000.
e. List all information from the Sales_Order table for orders placed in the month of June.
f. List the order information for ClientNo 'C00001' and 'C00002'.
g. List products whose selling price is greater than 500 and less than or equal to 750.
h. List products whose selling price is more than 500. Calculate a new selling price as, original selling price * .15. Rename the new column in the output of the above query as new_price.
i. List the names, city and state of clients who are not in the state of 'Maharashtra'.
j. Count the total number of orders.
k. Calculate the average price of all the products.
l. Determine the maximum and minimum product prices. Rename the output as max_price and min_price respectively.
m. Count the number of products having price less than or equal to 500.
n. List all the products whose QtyOnHand is less than reorder level.

2. Exercise on Date Manipulation:

a. List the order number and day on which clients placed their order.
b. List the month (in alphabets) and date when the orders must be delivered.
c. List the OrderDate in the format 'DD-Month-YY'. e.g. 12-February-02.
d. List the date, 15 days after today's date.

13. Unions can be used in subqueries.

14. The Intersect clause outputs only rows produced by both the queries intersected.

15. The Minus clause outputs the rows produced by the first query, before filtering the rows retrieved by the second query.

HANDS ON EXERCISES

1. **Exercises on using Having and Group By Clauses:**

a. Print the description and total qty sold for each product.
b. Find the value of each product sold.
c. Calculate the average qty sold for each client that has a maximum order value of 15000.00.
d. Find out the total of all the billed orders for the month of June.

2. **Exercises on Joins and Correlation:**

a. Find out the products, which have been sold to 'Ivan Bayross'.
b. Find out the products and their quantities that will have to be delivered in the current month.
c. List the ProductNo and description of constantly sold (i.e. rapidly moving) products.
d. Find the names of clients who have purchased 'Trousers'.
e. List the products and orders from customers who have ordered less than 5 units of 'Pull Overs'.
f. Find the products and their quantities for the orders placed by 'Ivan Bayross' and 'Mamta Muzumdar'.
g. Find the products and their quantities for the orders placed by ClientNo 'C00001' and 'C00002'.

3. **Exercise on Sub-queries:**

a. Find the ProductNo and description of non-moving products i.e. products not being sold.
b. List the customer Name, Address1, Address2, City and PinCode for the client who has placed order no 'O19001'.
c. List the client names that have placed orders before the month of May'02.
d. List if the product 'Lycra Top' has been ordered by any client and print the Client_no, Name to whom it was sold.
e. List the names of clients who have placed orders worth Rs. 10000 or more.

HANDS ON EXERCISES

1. Write appropriate SQL statements for the following:

a) Create a simple index idx_Prod on product cost price from the Product_Master table.

b) Create a sequence inv_seq with the following parameters,
increment by 3, cycle, cache 4 and which will generate the numbers from 1 to 9999 in ascending order.

c) Create view on OrderNo, OrderDate, OrderStatus of the Sales_Order table and ProductNo,ProductRate and QtyOrdered of Sales_Order_Details.

SOLUTIONS TO HANDS ON EXERCISES

7. INTERACTIVE SQL PART - I

1. SQL Statement for creating the tables:

a) **Table Name:** CLIENT_MASTER

```
CREATE TABLE CLIENT_MASTER(CLIENTNO varchar2(6), NAME varchar2(20),
    ADDRESS1 varchar2(30), ADDRESS2 varchar2(30), CITY varchar2(15),
    PINCODE number(8), STATE varchar2(15), BALDUE number(10,2));
```

b) **Table Name:** PRODUCT_MASTER

```
CREATE TABLE PRODUCT_MASTER(PRODUCTNO varchar2(6), DESCRIPTION varchar2(15),
    PROFITPERCENT number(4,2), UNITMEASURE varchar2(10), QTYONHAND number(8),
    REORDERLVL number(8), SELLPRICE number(8,2), COSTPRICE number(8,2));
```

c) **Table Name:** SALESMAN_MASTER

```
CREATE TABLE SALESMAN_MASTER(SALESMANNO varchar2(6),
    SALESMANNAME varchar2(20), ADDRESS1 varchar2(30), ADDRESS2 varchar2(30),
    CITY varchar2(20), PINCODE number(8), STATE varchar2(20), SALAMT number(8,2),
    TGTTOGET number(6,2), YTDSALES number(6,2), REMARKS varchar2(60));
```

2. SQL Statement for inserting into their respective tables:

a) Data for **CLIENT_MASTER** table:

```
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00001', 'Ivan Bayross', 'Mumbai', 400054, 'Maharashtra', 15000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00002', 'Mamta Muzumdar', 'Madras'', 780001, 'Tamil Nadu', 0);
INSERT INTO Client_Master (ClientNo, Name, City, Pincode, State, BalDue)
    VALUES ('C00003', 'Chhaya Bankar', 'Mumbai', 400057, 'Maharashtra', 5000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00004', 'Ashwini Joshi', 'Bangalore', 560001, 'Karnataka', 0);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00005', 'Hansel Colaco', 'Mumbai', 400060, 'Maharashtra', 2000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00006', 'Deepak Sharma', 'Mangalore', 560050, 'Karnataka', 0);
```

b) Data for **PRODUCT_MASTER** table

```
INSERT INTO Product_Master VALUES ('P00001', 'T-Shirts', 5, 'Piece', 200, 50, 350, 250);
INSERT INTO Product_Master VALUES ('P03453', 'Shirts', 6, 'Piece', 150, 50, 500, 350);
INSERT INTO Product_Master VALUES ('P06734', 'Cotton Jeans', 5, 'Piece', 100, 20, 600, 450);
INSERT INTO Product_Master VALUES ('P07865', 'Jeans', 5, 'Piece', 100, 20, 750, 500);
INSERT INTO Product_Master VALUES ('P07868', 'Trousers', 2, 'Piece', 150, 50, 850, 550);
INSERT INTO Product_Master VALUES ('P07885', 'Pull Overs', 2.5, 'Piece', 80, 30, 700, 450);
INSERT INTO Product_Master VALUES ('P07965', 'Denim Shirts', 4, 'Piece', 100, 40, 350, 250);
INSERT INTO Product_Master VALUES ('P07975', 'Lycra Tops', 5, 'Piece', 70, 30, 300, 175);
INSERT INTO Product_Master VALUES ('P08865', 'Skirts', 5, 'Piece', 75, 30, 450, 300);
```

c) Data for **SALESMAN_MASTER** table

```
INSERT INTO Salesman_Master VALUES ('S00001', 'Aman', 'A/14', 'Worli', 'Mumbai', 400002,
                                    'Maharashtra', 3000, 100, 50, 'Good');
INSERT INTO Salesman_Master VALUES ('S00002', 'Omkar', '65', 'Nariman', 'Mumbai', 400001,
                                    'Maharashtra', 3000, 200, 100, 'Good');
INSERT INTO Salesman_Master VALUES ('S00003', 'Raj', 'P-7', 'Bandra', 'Mumbai', 400032,
                                    'Maharashtra', 3000, 200, 100, 'Good');
INSERT INTO Salesman_Master VALUES ('S00004', 'Ashish', 'A/5', 'Juhu', 'Bombay', 400044,
                                    'Maharashtra', 3500, 200, 150, 'Good');
```

3. SQL Statement for retrieving records from a table:

a) Find out the names of all the clients.
```
SELECT Name FROM Client_Master;
```
b) Retrieve the entire contents of the Client_Master table.
```
SELECT * FROM Client_Master;
```
c) Retrieve the list of names, city and the sate of all the clients.
```
SELECT Name, City, State FROM Client_Master;
```
d) List the various products available from the Product_Master table.
```
SELECT Description FROM Product_Master;
```
e) List all the clients who are located in Mumbai.
```
SELECT * FROM Client_Master WHERE City = 'Mumbai';
```
f) Find the names of salesmen who have a salary equal to Rs.3000.
```
SELECT Salesman_name FROM Salesman_Master WHERE SalAmt = 3000;
```

4. SQL Statement for updating records in a table:

a) Change the city of ClientNo 'C00005' to 'Bangalore'.
```
UPDATE Client_Master SET City = 'Bangalore' WHERE ClientNo = 'C00005';
```
b) Change the BalDue of ClientNo 'C00001' to Rs. 1000.
```
UPDATE Client_Master SET BalDue = 1000 WHERE Client_no = 'C00001';
```
c) Change the cost price of 'Trousers' to Rs. 950.00.
```
UPDATE Product_Master SET CostPrice = 950.00 WHERE Description = 'Trousers';
```
d) Change the city of the salesman to Pune.
```
UPDATE Client_Master SET City = 'Pune';
```

5. SQL Statement for deleting records in a table:

a) Delete all salesmen from the Salesman_Master whose salaries are equal to Rs. 3500.
```
DELETE FROM Salesman_Master WHERE SalAmt = 3500;
```
b) Delete all products from Product_Master where the quantity on hand is equal to 100.
```
DELETE FROM Product_Master WHERE QtyOnHand = 100;
```
c) Delete from Client_Master where the column state holds the value 'Tamil Nadu'.
```
DELETE FROM Client_Master WHERE State = 'Tamil Nadu';
```

6. SQL Statement for altering the table structure:

a) Add a column called 'Telephone' of data type 'number' and size ='10' to the Client_Master table.
```
ALTER TABLE Client_Master ADD (Telephone number(10));
```
b) Change the size of SellPrice column in Product_Master to 10,2.
```
ALTER TABLE Product_Master MODIFY (SellPrice number(10,2));
```

7. SQL Statement for deleting the table structure along with the data:

a) Destroy the table Client_Master along with its data.
```
DROP TABLE Client_Master;
```

8. SQL Statement for renaming the table:

a) Change the name of the Salesman_Master table to sman_mast.

```
RENAME Salesman_Master TO sman_mast;
```

8. INTERACTIVE SQL PART - II

1. SQL Statement for creating the tables:

Note

Before, executing the CREATE TABLE with the Data Constraints ensure that the tables with similar names do not exist within the database. Executing the following SQL commands will eliminate the problem of existing tables.

```
DROP TABLE IF EXISTS SALES_ORDER_DETAILS;
DROP TABLE IF EXISTS SALES_ORDER;
DROP TABLE IF EXISTS SALESMAN_MASTER;
DROP TABLE IF EXISTS PRODUCT_MASTER;
DROP TABLE IF EXISTS CLIENT_MASTER;
```

a) **Table Name:** CLIENT_MASTER

```
CREATE TABLE CLIENT_MASTER(CLIENTNO varchar2(6) PRIMARY KEY,
    NAME varchar2(20) NOT NULL, ADDRESS1 varchar2(30), ADDRESS2 varchar2(30),
    CITY varchar2(15), PINCODE number(8), STATE varchar2(15), BALDUE number(10,2),
    CONSTRAINT ck_client CHECK (CLIENTNO like 'C%'));
```

b) **Table Name:** PRODUCT_MASTER

```
CREATE TABLE PRODUCT_MASTER(PRODUCTNO varchar2(6) PRIMARY KEY,
    DESCRIPTION varchar2(15) NOT NULL, PROFITPERCENT number(4,2) NOT NULL,
    UNITMEASURE varchar2(10) NOT NULL, QTYONHAND number(8) NOT NULL,
    REORDERLVL number(8) NOT NULL, SELLPRICE number(8,2) NOT NULL,
    COSTPRICE number(8,2) NOT NULL,
    CONSTRAINT ck_product CHECK (PRODUCTNO like 'P%'),
    CONSTRAINT ck_sell CHECK (SELLPRICE <> 0),
    CONSTRAINT ck_cost CHECK (COSTPRICE <> 0));
```

c) **Table Name:** SALESMAN_MASTER

```
CREATE TABLE SALESMAN_MASTER(SALESMANNO varchar2(6) PRIMARY KEY,
    SALESMANNAME varchar2(20) NOT NULL, ADDRESS1 varchar2(30) NOT NULL,
    Address2 varchar2(30), CITY varchar2(20), PINCODE number(8), State varchar2(20),
    SALAMT number(8,2) NOT NULL, TGTTOGET number(6,2) NOT NULL,
    YTDSALES number(6,2) NOT NULL, REMARKS varchar2(60),
    CONSTRAINT ck_salesman CHECK (SALESMANNO like 'S%'),
    CONSTRAINT ck_sal CHECK (SALAMT <> 0),
    CONSTRAINT ck_target CHECK (TGTTOGET <> 0));
```

d) **Table Name:** SALES_ORDER

```
CREATE TABLE SALES_ORDER(ORDERNO  varchar2(6) PRIMARY KEY,
  CLIENTNO varchar2(6) REFERENCES CLIENT_MASTER, ORDERDATE date,
  DELYADDR varchar2(25), SALESMANNO varchar2(6) REFERENCES SALESMAN_MASTER,
  DELYTYPE char(1) DEFAULT 'F', BILLEDYN char(1), DELYDATE date,
  ORDERSTATUS varchar2(10), CONSTRAINT ck_order CHECK (ORDERNO like 'O%'),
  CONSTRAINT ck_dely_type CHECK (DELYTYPE IN ('P', 'F')),
  CONSTRAINT ck_ord_status
     CHECK(ORDERSTATUS IN ('In Process', 'Fulfilled', 'Backorder', 'Cancelled')));
```

e) **Table Name:** SALES_ORDER_DETAILS

```
CREATE TABLE SALES_ORDER_DETAILS(
    ORDERNO varchar2(6) REFERENCES SALES_ORDER,
    PRODUCTNO varchar2(6) REFERENCES PRODUCT_MASTER,
    QTYORDERED number(8), QTYDISP number(8), PRODUCTRATE number(10,2),
    PRIMARY KEY (ORDERNO, PRODUCTNO));
```

2. SQL Statement for inserting data into their respective tables:

a) Data for **CLIENT_MASTER** table:

```
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00001', 'Ivan Bayross', 'Mumbai', 400054, 'Maharashtra', 15000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00002', 'Mamta Muzumdar', 'Madras'', 780001, 'Tamil Nadu', 0);
INSERT INTO Client_Master (ClientNo, Name, City, Pincode, State, BalDue)
    VALUES ('C00003', 'Chhaya Bankar', 'Mumbai', 400057, 'Maharashtra', 5000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00004', 'Ashwini Joshi', 'Bangalore', 560001, 'Karnataka', 0);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00005', 'Hansel Colaco', 'Mumbai', 400060, 'Maharashtra', 2000);
INSERT INTO Client_Master (ClientNo, Name, City, PinCode, State, BalDue)
    VALUES ('C00006', 'Deepak Sharma', 'Mangalore', 560050, 'Karnataka', 0);
```

b) Data for **PRODUCT_MASTER** table

```
INSERT INTO Product_Master VALUES ('P00001', 'T-Shirts', 5, 'Piece', 200, 50, 350, 250);
INSERT INTO Product_Master VALUES ('P03453', 'Shirts', 6, 'Piece', 150, 50, 500, 350);
INSERT INTO Product_Master VALUES ('P06734', 'Cotton Jeans', 5, 'Piece', 100, 20, 600, 450);
INSERT INTO Product_Master VALUES ('P07865', 'Jeans', 5, 'Piece', 100, 20, 750, 500);
INSERT INTO Product_Master VALUES ('P07868', 'Trousers', 2, 'Piece', 150, 50, 850, 550);
INSERT INTO Product_Master VALUES ('P07885', 'Pull Overs', 2.5, 'Piece', 80, 30, 700, 450);
INSERT INTO Product_Master VALUES ('P07965', 'Denim Shirts', 4, 'Piece', 100, 40, 350, 250);
INSERT INTO Product_Master VALUES ('P07975', 'Lycra Tops', 5, 'Piece', 70, 30, 300, 175);
INSERT INTO Product_Master VALUES ('P08865', 'Skirts', 5, 'Piece', 75, 30, 450, 300);
```

c) Data for **SALESMAN_MASTER** table

```
INSERT INTO Salesman_Master VALUES ('S00001', 'Aman', 'A/14', 'Worli', 'Mumbai', 400002,
                                    'Maharashtra', 3000, 100, 50, 'Good');
INSERT INTO Salesman_Master VALUES ('S00002', 'Omkar', '65', 'Nariman', 'Mumbai', 400001,
                                    'Maharashtra', 3000, 200, 100, 'Good');
INSERT INTO Salesman_Master VALUES ('S00003', 'Raj', 'P-7', 'Bandra', 'Mumbai', 400032,
                                    'Maharashtra', 3000, 200, 100, 'Good');
INSERT INTO Salesman_Master VALUES ('S00004', 'Ashish', 'A/5', 'Juhu', 'Bombay', 400044,
                                    'Maharashtra', 3500, 200, 150, 'Good');
```

d) Data for **SALES_ORDER** table

```
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O19001', '12-june-02', 'C00001', 'F', ' N', 'S00001', '20-july-02', 'In Process');
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O19002', '25-june-02', 'C00002', 'P', ' N', 'S00002', '27-july-02', 'Cancelled');
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O19003', '18-feb-02', 'C00003', 'F', ' Y', 'S00003', '20-feb-02', 'Fulfilled');
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O19003', '03-apr-02', 'C00001', 'F', 'Y', 'S00001', '07-apr-02', 'Fulfilled');
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O46866', '20-may-02', 'C00004', 'P', 'N', 'S00002', '22-may-02', 'Cancelled');
```

```
INSERT INTO Sales_Order (OrderNo, OrderDate, ClientNo, DelyType, BilledYn, SalesmanNo, DelyDate,
    OrderStatus) VALUES('O19008', '24-may-02', 'C00005', 'F', 'N', 'S00004', '26-july-96', 'In Process');
```

e) Data for **SALES_ORDER_DETAILS** table

```
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19001', 'P00001', 4, 4, 525);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19001', 'P07965', 2, 1, 8400);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19001', 'P07885', 2, 1, 5250);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19002', 'P00001', 10, 0, 525);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46865', 'P07868', 3, 3, 3150);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46865', 'P07885', 3, 1, 5250);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46865', 'P00001', 10, 10, 525);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46865', 'P03453', 4, 4, 1050);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19003', 'P03453', 2, 2, 1050);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19003', 'P06734', 1, 1, 12000);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46866', 'P07965', 1, 0, 8400);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O46866', 'P07975', 1, 0, 1050);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19008', 'P00001', 10, 5, 525);
INSERT INTO Sales_Order_Details (OrderNo, ProductNo, QtyOrdered, QtyDisp, ProductRate)
    VALUES('O19008', 'P07975', 5, 3, 1050);
```

9. INTERACTIVE SQL PART - III

1. Generate SQL Statements to perform the following computations on table data:

a. Listing of the names of all clients having 'a' as the second letter in their names.
```
SELECT Name FROM Client_Master WHERE Name like '_a%';
```
b. Listing of clients who stay in a city whose first letter is 'M'.
```
SELECT ClientNo, Name FROM Client_Master WHERE City LIKE 'M%';
```
c. List all clients who stay in 'Bangalore' or 'Mangalore'
```
SELECT ClientNo, Name FROM Client_Master WHERE City IN('Bangalore', 'Mangalore');
```
d. List all clients whose BalDue is greater than value 10000.
```
SELECT ClientNo, Name FROM Client_Master WHERE Baldue > 10000;
```
e. Print the information from Sales_Order table for orders placed in the month of June.
```
SELECT * FROM Sales_Order WHERE TO_CHAR(OrderDate,'MON') = 'JUN';
```
f. Displaying the order information of ClientNo 'C00001' and 'C00002'.
```
SELECT * FROM Sales_Order WHERE ClientNo IN('C00001', 'C00002');
```
g. List products whose selling price is greater than 500 and less than or equal to 750.
```
SELECT ProductNo, Description FROM Product_Master WHERE SellPrice > 500 and SellPrice < 750;
```

h. Listing of products whose selling price is more than 500 with the new selling price calculated as original selling price plus 15%.
```
SELECT ProductNo, Description, SellPrice, SellPrice*15 new_price FROM Product_Master
    WHERE SellPrice > 500;
```
i. Listing of names, city and state of clients who are not in the state of 'Maharashtra'.
```
SELECT Name, City, State FROM Client_Master WHERE State NOT IN('Maharashtra');
```
j. Count the total number of orders.
```
SELECT COUNT(OrderNo) 'No. Of Order' FROM Sales_Order;
```
k. Calculating the average price of all the products.
```
SELECT AVG(SellPrice) FROM Product_Master;
```
l. Determining the maximum and minimum price for the product prices.
```
SELECT MAX(SellPrice) max_price, MIN(SellPrice) min_price FROM Product_Master;
```
m. Count the number of products having price greater than or equal to 500.
```
SELECT COUNT(ProductNo) FROM Product_Master WHERE SellPrice <= 1500;
```
n. Find all the products whose QtyOnHand is less than reorder level.
```
SELECT ProductNo, Description FROM Product_Master WHERE QtyOnHand < ReorderLvl;
```

2. SQL Statements for Date Manipulation:

a. Display the order number and day on which clients placed their order.
```
SELECT OrderNo, TO_CHAR(OrderDate, 'day') FROM Sales_Order;
```
b. Display the month (in alphabets) and date when the order must be delivered.
```
SELECT TO_CHAR(DelyDate, 'month'), DelyDate FROM Sales_Order
    ORDER BY TO_CHAR(DelyDate, 'month');
```
c. Display the OrderDate in the format 'DD-Month-YY'. E.g. 12-February-03
```
SELECT DATE_FORMAT(OrderDATE '%d-%M-%Y') FROM Sales_Order;
```
d. List the OrderDate in the format 'DD-Month-YY'. e.g. 12-February-02.
```
SELECT TO_CHAR(Orderdate, 'DD-Month-YY') FROM Sales_Order;
```
e. Find the date, 15 days after today's date.
```
SELECT SYSDATE + 15 FROM DUAL;
```

10. INTERACTIVE SQL PART - IV

1. SQL statements for using Having and Group By Clauses:

a. Printing the description and total quantity sold for each product.
```
SELECT description, SUM(QtyDisp) FROM Product_Master, Sales_Order_Details
    WHERE Product_Master.ProductNo = Sales_Order_Details.ProductNo
    GROUP BY Description;
```
b. Finding the value of each product sold.
```
SELECT Sales_Order_Details.ProductNo, Product_Master.Description,
    SUM(Sales_Order_Details.QtyDisp * Sales_Order_Details.ProductRate) 'Sales Per Product'
    FROM Sales_Order_Details, Product_Master
        WHERE Product_Master.ProductNo = Sales_Order_Details.ProductNo
        GROUP BY Sales_Order_Details.ProductNo, Product_Master.Description;
```
c. Calculating the average quantity sold for each client that has a maximum order value of 15000.00.
```
SELECT CM.ClientNo, CM.Name, AVG(SOD.QtyDisp) 'Avg. Sales'
    FROM Sales_Order_Details SOD, Sales_Order SO, Client_Master CM
        WHERE CM.ClientNo = SO.ClientNo AND SO.OrderNo = SOD.OrderNo
    GROUP BY CM.ClientNo, Name HAVING MAX(SOD.QtyOrdered * SOD.ProductRate) > 15000;
```
d. Finding out the total of all the billed orders for the month of June.
```
SELECT SO.OrderNo, SO.OrderDate, SUM(SOD.QtyOrdered * SOD.ProductRate) 'Order Billed'
    FROM Sales_Order SO, Sales_Order_Details SOD WHERE SOD.OrderNo = SO.OrderNo
    AND SO.Billed = 'Y' AND to_char(OrderDate, 'MON') = 'Jun' GROUP BY SO.OrderNo;
```

2. Exercises on Joins and Correlation:

a. Find out the products, which have been sold to 'Ivan Bayross'.

```
SELECT SOD.ProductNo, PM.Description
    FROM Sales_Order_Details SOD, Sales_Order SO, Product_Master PM, Client_Master CM
        WHERE PM.ProductNo = SOD.ProductNo AND SO.OrderNo = SOD.OrderNo
            AND CM.ClientNo = SO.ClientNo AND CM.Name = 'Ivan Bayross';
```

b. Finding out the products and their quantities that will have to be delivered in the current month.

```
SELECT SOD.ProductNo, PM.Description, SUM(SOD.QtyOrdered)
    FROM Sales_Order_Details SOD, Sales_Order SO, Product_Master PM
        WHERE PM.ProductNo = SOD.ProductNo AND SO.OrderNo = SOD.OrderNo
            AND TO_CHAR(DelyDate, 'MON-YY') = TO_CHAR(SYSDATE, 'MON-YY')
        GROUP BY SOD.ProductNo, PM.Description;
```

c. Listing the ProductNo and description of constantly sold (i.e. rapidly moving) products.

```
SELECT DISTINCT Product_Master.ProductNo, Description
    FROM Sales_Order_Details, Product_Master
        WHERE Product_Master.ProductNo =Sales_Order_Details.ProductNo;
```

d. Finding the names of clients who have purchased 'Trousers'.

```
SELECT DISTINCT Sales_Order.ClientNo, Client_Master.Name
    FROM Sales_Order_Details, Sales_Order, Product_Master, Client_Master
        WHERE Product_Master.ProductNo = Sales_Order_Details.ProductNo
            AND Sales_Order.OrderNo = Sales_Order_Details.OrderNo
            AND Client_Master.ClientNo = Sales_Order.ClientNo
            AND Description = 'Trousers';
```

e. Listing the products and orders from customers who have ordered less than 5 units of 'Pull Overs'.

```
SELECT Sales_Order_Details.ProductNo, Sales_Order_Details.OrderNo
    FROM Sales_Order_Details, Sales_Order, Product_Master
        WHERE Sales_Order.OrderNo = Sales_Order_Details.OrderNo
        AND Product_Master.ProductNo = Sales_Order_Details.ProductNo
        AND Sales_Order_Details.QtyOrdered < 5 AND Product_Master.Description = 'Pull Overs';
```

f. Finding the products and their quantities for the orders placed by 'Ivan Bayross' and 'Mamta Muzumdar'.

```
SELECT SOD.ProductNo, PM.Description, SUM(QtyOrdered) 'Units Ordered'
    FROM Sales_Order_Details SOD, Sales_Order SO, Product_Master PM, Client_Master CM
        WHERE SO.OrderNo = SOD.OrderNo AND PM.ProductNo = SOD.ProductNo
            AND CM.ClientNo = SO.ClientNo
            AND (CM.Name = 'Ivan Bayross' OR CM.Name = 'Mamta Muzumdar')
        GROUP BY SOD.ProductNo, PM.Description;
```

g. Finding the products and their quantities for the orders placed by ClientNo 'C00001' and 'C00002'.

```
SELECT SO.ClientNo, SOD.ProductNo, PM.Description, SUM(QtyOrdered) 'Units Ordered'
    FROM Sales_Order SO, Sales_Order_Details SOD, Product_Master PM, Client_Master CM
        WHERE SO.OrderNo = SOD.OrderNo AND SOD.ProductNo = PM.ProductNo
            AND SO.ClientNo = CM.ClientNo
        GROUP BY SO.ClientNo, SOD.ProductNo, PM.Description
            HAVING SO.ClientNo = 'C00001' OR SO.ClientNo='C00002';
```

3. SQL statements for exercises on Sub-queries:

a. Finding the non-moving products i.e. products not being sold.

```
SELECT ProductNo, Description FROM Product_Master
    WHERE ProductNo NOT IN(SELECT ProductNo FROM Sales_Order_Details);
```

b. Finding the name and complete address for the customer who has placed Order number 'O19001'.

```
SELECT Name ,Address1, Address2, City, State, PinCode FROM Client_Master
    WHERE ClientNo IN(SELECT ClientNo FROM Sales_Order WHERE OrderNo = 'O19001');
```

c. Finding the clients who have placed orders before the month of May'02.

```
SELECT ClientNo, Name FROM Client_Master WHERE ClientNo IN(SELECT ClientNo
    FROM Sales_Order WHERE TO_CHAR(OrderDate, 'MON,YY') < 'MAY,02');
```

d. Find out if the product 'Lycra Tops' has been ordered by any client and print the ClientNo, Name to whom it was sold.

```
SELECT ClientNo, Name FROM Client_Master WHERE ClientNo
    IN(SELECT ClientNo FROM Sales_Order WHERE OrderNo IN(SELECT OrderNo
        FROM Sales_Order_Details WHERE ProductNo IN(SELECT ProductNo
            FROM Product_Master WHERE Description = 'Lycra Tops')));
```

e. Find the names of clients who have placed orders worth Rs. 10000 or more.

```
SELECT Name FROM Client_Master WHERE ClientNo IN(SELECT ClientNo FROM Sales_Order
    WHERE OrderNo IN(SELECT OrderNo FROM Sales_Order_Details
        WHERE (QtyOrdered * ProductRate) >= 10000));
```

SOLUTIONS TO HANDS ON EXERCISES

11. SQL PERFORMANCE TUNING

1. Writing appropriate SQL statements for the following:

a) Creating a simple index idx_Prod on product cost price from the Product_Master table.
```
CREATE INDEX idx_prod ON Product_Master (CostPrice);
```

b) Creating a sequence inv_seq.
```
CREATE SEQUENCE inv_seq
    INCREMENT BY 3 START WITH 1
    MINVALUE 1 MAXVALUE 9999 CYCLE Cache 4;
```

c) Create view on OrderNo, OrderDate, OrderStatus of the Sales_Order table and ProductNo,ProductRate and QtyOrdered of Sales_Order_Details.
```
CREATE VIEW vw_Sal_Ord AS
    SELECT s.OrderNo, s.OrderDate, sod.ProductNo, sod.ProductRate, sod.QtyOrdered,
           s.Orderstatus
    FROM Sales_Order s, Sales_Order_Details sod WHERE s.OrderNo = sod.OrderNo;
```

12. SECURITY MANAGEMENT USING SQL

1. Giving the user IVAN permission only to view records in the tables Sales_order and Sales_order_details, along with an option to further grant permission on these tables to other users.
```
GRANT SELECT ON Sales_Order TO IVAN WITH GRANT OPTION;
GRANT SELECT ON Sales_Order_Details TO IVAN WITH GRANT OPTION;
```

2. Giving the user IVAN all data manipulation privileges on the table Client_Master without an option to further grant permission on the Client_Master table to other users.
```
GRANT ALL ON Client_Master TO IVAN;
```

3. Taking back all privileges given to the user IVAN on the table Client_Master.
```
REVOKE ALL ON Client_Master FROM IVAN;
```

13. OOPS IN ORACLE

1. **Creatint the Address_Ty type:**

a) **Type Name**: Address_Ty
```
CREATE TYPE Address_Ty AS OBJECT (
    Address1 varchar2(30), Address2 varchar2(30), City varchar2(20),
    PinCode number, State varchar2(20));
```