SE-Computer A Batch C	Roll number : 9913
Experiment no. : 3(Part-1)	Date of Implementation :

Aim : To implement data definition language (DDL) commands

Tool Used : PostgreSQL

Related Course outcome : At the end of the course, Students will be able to Use SQL : Standard language of relational database

Rubrics for assessment of Experiment:

Indicator	Poor	Average	Good
Timeliness ● Maintains assignment deadline (3)	Assignment not done (0)	One or More than One week late (1-2)	Maintains deadline (3)
Completeness and neatness ● Complete all parts of assignment(3)	N/A	< 80% complete (1-2)	100% complete (3)
Originality ● Extent of plagiarism(2)	Copied it from someone else(0)	At least few questions have been done without copying(1)	Assignment has been solved completely without copying (2)
Knowledge ● In depth knowledge of the assignment(2)	Unable to answer 2 questions(0)	Unable to answer 1 question (1)	Able to answer 2 questions (2)

Assessment Marks :

Timeliness	
Completeness and neatness	
Originality	
Knowledge	
Total	

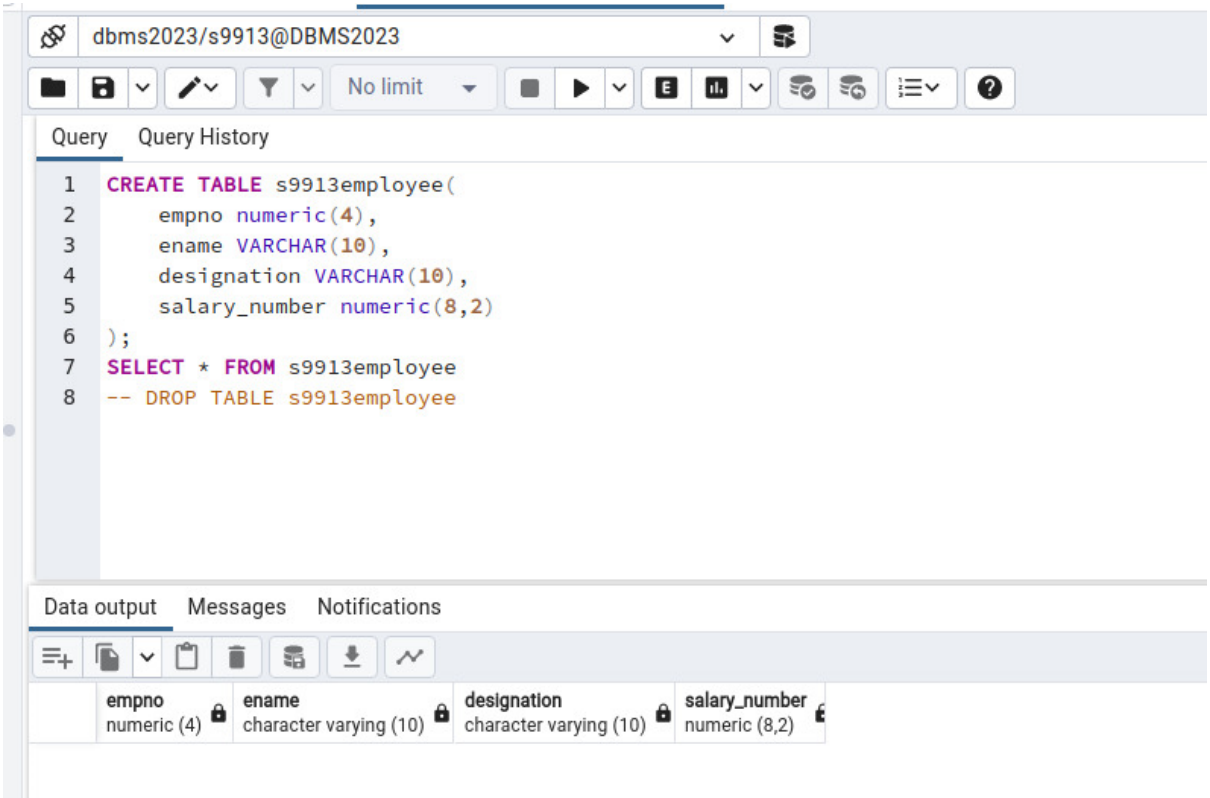
Total : (Out of 10)

Teacher's Sign :

	EXPERIMENT 3	DDL Commands
	Aim	To implement DDL – Data definition language command
	Tools	PostgreSQL/MYSQL
	Theory	**SQL:** It is structured query language, basically used to pass the query to retrieve and manipulate the information from database **DDL:** The Data Definition Language (DDL) is used to create the database (i.e. tables, keys, relationships etc), maintain the structure of the database and destroy databases and database objects. Eg. Create, Drop, Alter, Describe, Truncate 1. **CREATE statements: It is used to create the table.** CREATE TABLE table_name(columnName1 datatype(size), columnName2 datatype(size),……); 2. **DROP statements:** To destroy an existing database, table, index, or view. If a table is dropped all records held within it are lost and cannot be recovered. DROP TABLE table_name; 3. **ALTER statements:** To modify an existing database object. **Adding new columns:** Alter table table_name Add(New_columnName1 datatype(size), New_columnName2 datatype(size),……); **Dropping a columns from a table** : Alter table table_name DROP column columnName: **Modifying Existing columns:** Alter table table_name Modify (columnName1 Newdatatype(Newsize)); 4. **Describe statements:** To describe the structure (column and data types) of an existing database, table, index, or view. DESC table_name; 5. **Truncate statements:** To destroy the data in an existing database, table, index, or view. If a table is truncated all records held within it are lost and cannot be recovered but the table structure is maintained. TRUNCATE TABLE table_name;

Procedure	1. Write a query to create a table employee with empno, ename, designation, and salary. Emp (empno number (4), ename varchar (10), designation varchar (10), salary number (8,2)); 2. Write a Query to Alter the column empno number (4) to empno number (6). 3. Write a Query to Alter the table employee with multiple columns (empno, ename.) 4. Write a query to add a new column in to employee as qualification varchar2(6) 5. Write a query to add multiple columns in to employee dob date , doj date 6. Write a query to drop a column 'doj' from an existing table employee 7. Write a query to drop multiple columns 'dob' and 'qualification' from employee 8. Truncate table EMP 9. Drop table EMP
Post Lab Questions:	1. What is Data Dictionary? 2. What is Schema? 3. What are different data types in SQL?

1



2

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
alter column empno type
numeric(6);

SELECT * FROM s9913employee
```

Data output Messages Notifications

empno numeric (6)	ename character varying (10)	designation character varying (10)	salary_number numeric (8,2)

3

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
alter column empno type
numeric(10),
alter column ename type
varchar(20);
SELECT * FROM s9913employee
```

Data output Messages Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)

4

dbms2023/s9913@DBMS2023

No limit

Query | Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
add column qualification varchar(6);
SELECT * FROM s9913employee
```

Data output | Messages | Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)	qualification character varying (6)

5

dbms2023/s9913@DBMS2023

No limit

Query | Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
add column dob date,
add column doj date;
SELECT * FROM s9913employee
```

Data output | Messages | Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)	qualification character varying (6)	dob date	doj date

6

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
drop column doj;
SELECT * FROM s9913employee
```

Data output Messages Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)	qualification character varying (6)	dob date

7

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
ALTER TABLE s9913employee
drop column dob,
drop column qualification;
SELECT * FROM s9913employee
```

Data output Messages Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)

8

dbms2023/s9913@DBMS2023

No limit

Query | Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
insert into s9913employee
values(1,'Mark','HR',300000.00);
-- truncate table s9913employee;
SELECT * FROM s9913employee
```

Data output | Messages | Notifications

	empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)
1	1	Mark	HR	300000.00

dbms2023/s9913@DBMS2023

No limit

Query | Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- DROP TABLE s9913employee
-- insert into s9913employee
-- values(1,'Mark','HR',300000.00);
truncate table s9913employee;
SELECT * FROM s9913employee
```

Data output | Messages | Notifications

empno numeric (10)	ename character varying (20)	designation character varying (10)	salary_number numeric (8,2)

dbms2023/s9913@DBMS2023

No limit

Query | Query History

```
-- CREATE TABLE s9913employee(
--  empno numeric(4),
--  ename VARCHAR(10),
--  designation VARCHAR(10),
--  salary_number numeric(8,2)
-- )
-- insert into s9913employee
-- values(1,'Mark','HR',300000.00);
-- truncate table s9913employee;
DROP TABLE s9913employee
-- SELECT * FROM s9913employee
```

Data output | Messages | Notifications

```
DROP TABLE

Query returned successfully in 234 msec.
```

POSTLAB:-

Q1

A data dictionary is a centralized repository that stores metadata about a database, including definitions, data types, constraints, relationships, and other details. It serves as a reference guide for understanding and managing the structure and attributes of data within the database.

Q2

In a database, a schema is a logical container or namespace that holds a collection of database objects, including tables, views, indexes, and procedures. It provides a way to organize and manage database objects, allowing multiple users or applications to work independently within their designated schemas. Schemas help avoid naming conflicts and provide a structure for organizing and securing database elements.

Q3

Numeric Types:

INT, INTEGER: Integer.
SMALLINT: Small integer.
TINYINT: Very small integer.
BIGINT: Large integer.
DECIMAL(p, s), NUMERIC(p, s): Decimal number with a specified precision (p) and scale (s).
FLOAT: Floating-point number.
REAL: Real number.

Character/String Types:

CHAR(n): Fixed-length character string.
VARCHAR(n), VARCHAR(MAX): Variable-length character string with a maximum length of n characters or maximum allowed length.
TEXT: Variable-length character string with no specified maximum length.

Date and Time Types:

DATE: Date (year, month, day).
TIME: Time of day.
DATETIME, TIMESTAMP: Date and time.
INTERVAL: Time interval.

Boolean Type:

BOOLEAN, BOOL: Boolean

SEcomputer A batch-C	Roll number : 9913
Experiment no. : 3 part2	Date of Implementation :
Aim : To implement data manipulation language (DML) commands	
Tool Used : PostgreSQL	
Related Course outcome : Students should be able to Write queries in SQL to retrieve any type of information from a data base.	

Rubrics for assessment of Experiment:

Indicator	Poor	Average	Good
Timeliness Maintains Experiment deadline (3)	Experiment not done (0)	One or More than One week late (1-2)	Maintains deadline (3)
Completeness and neatness Complete all parts of Experiment(3)	N/A	< 80% complete (1-2)	100% complete (3)
Originality Extent of plagiarism(2)	Copied it from someone else(0)	At least try to implement but could not succeed (1)	Implemented (2)
Knowledge In depth knowledge of the Experiment(2)	Unable to answer any questions(0)	Unable to answer few questions (1)	Able to answer all questions (2)

Assessment Marks :

Timeliness	
Completeness and neatness	
Originality	
Knowledge	
Total	

Total : (Out of 10)

Teacher's Sign :	
EXPERIMENT 3	**DDL and DML Commands**
Aim	To implement DDL with integrity constraints and DML – Data manipulation language command
Tools	PostgreSQL/MySql
Theory	Data Definition Language-1) Create 2) Alter 3) Drop 4) Rename 5) Truncate ● **CREATE** – is used to create the database or its objects (like table, index, function, views, store procedure and triggers). ● **DROP** – is used to delete objects from the database. ● **ALTER**-is used to alter the structure of the database. ● **TRUNCATE**–is used to remove all records from a table, including all spaces allocated for the records are removed. ● **COMMENT** –is used to add comments to the data dictionary. ● **RENAME** –is used to rename an object existing in the database. 1) Create table create table tablename (column1 data type, column2 data type, column3 data type, ... columnN data type); 2) **DROP object object_name** Examples: DROP TABLE table_name; table_name: Name of the table to be deleted. DROP DATABASE database_name; database_name: Name of the database to be deleted.

3) TRUNCATE

TRUNCATE statement is a Data Definition Language (DDL) operation that is used to mark the extents of a table for deallocation (empty for reuse). The result of this operation quickly removes all data from a table, typically bypassing a number of integrity enforcing mechanisms. It was officially introduced in the standard.

The TRUNCATE TABLE mytable statement is logically (though not physically) equivalent to the DELETE FROM mytable statement (without a WHERE clause).
Syntax:
TRUNCATE TABLE table_name;
table_name: Name of the table to be truncated.
DATABASE name - student_data

- **cannot** be rolled back, so it must be used wisely.

DROP vs TRUNCATE

- Truncate is normally ultra-fast and its ideal for deleting data from a temporary table.
- Truncate preserves the structure of the table for future use, unlike drop table where the table is deleted with its full structure.

Table or Database deletion using DROP statement

- To delete the whole database
  DROP DATABASE student_data;

After running the above query whole database will be deleted.

- To truncate Student_details table from student_data database.
  TRUNCATE TABLE Student_details;

After running the above query Student_details table will be truncated, i.e, the data will be deleted but the structure will remain in the memory for further operations.

Alter

alter command is used for altering the table structure, such as,

- to add a column to existing table
- to rename any existing column
- to change data type of any column or to modify its size.
- to drop a column from the table.

```
ALTER TABLE table_name ADD(
   column_name datatype);
```

Procedure	**B)Data Manipulation Language** A Data Manipulation Language enables programmers and users of the database to retrieve insert, delete and update data in a database. e.g. INSERT, UPDATE, DELETE, SELECT. **<u>INSERT:</u>** INSERT statement adds one or more records to any single table in a relational database. INSERT INTO tablename VALUES (expr1,expr2……..); **<u>UPDATE:</u>** UPDATE statement that changes the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition. UPDATE table_name SET column_name = value [, column_name = value ...] [WHERE condition] **<u>DELETE:</u>** DELETE statement removes one or more records from a table. A subset may be defined for deletion using a condition, otherwise all records are removed. DELETE FROM tablename WHERE condition

Task1: 1. Create following tables:

Table name : client_master

Column Name	Data type	Size	
Client_no	varchar	6	Primary key
Name	varchar	20	Not null
Address	varchar	30	
City	varchar	15	
Pincode	numeric	8	
State	varchar	15	
Bal_due	numeric	10,2	>0

Table name: Product_master

Column Name	Data type	Size	
product_no	varchar	6	Primary key
description	varchar	15	Not null
Profit_percent	numeric	4,2	
Unit_measure	varchar	10	
Qty_on_hand	numeric	8	>0
Reorder_level	numeric	8	
Sell_price	numeric	8,2	
Cost_price	numeric	8,2	

2. Insert 5-6 records in each table.
3. Find out the names of all clients
4. Retrieve the entire contents of the client_master table.
5. Retrieve the list of names and cities of all the clients
6. List the various products available from the product_master table
7. List all the clients who are located in mumbai.
8. Change the city of client_no C001 to mumbai
9. Change the bal_due of client_no C005 to Rs. 1000
10. Change the cost price of 'hard disk' to Rs. 3000
11. Delete all the products from product_master where the qty_on_hand is less than 100
12. Delete from client_master where the column state holds the value 'Tamil Nadu'

Task2: Create the tables for EER diagram of EXPT. no 2

Post Lab Questions:	1. Explain different data types of Mysql/postgresql 2. Perform delete and truncate in lab and Differentiate delete and truncate

Q1

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
-- CREATE TABLE client_master (
--     client_no VARCHAR(6) PRIMARY KEY,
--     name VARCHAR(20) NOT NULL,
--     address VARCHAR(30),
--     city VARCHAR(15),
--     pincode NUMERIC(8),
--     state VARCHAR(15),
--     bal_due NUMERIC(10,2) CHECK (bal_due > 0)
-- );
select * from client_master
```

Data output Messages Notifications

client_no [PK] character varying (6)	name character varying (20)	address character varying (30)	city character varying (15)	pincode numeric (8)	state character varying (15)	bal_due numeric (10,2)

dbms2023/s9913@DBMS2023

No limit

Query Query History

```
CREATE TABLE Product_master (
    product_no VARCHAR(6) PRIMARY KEY,
    description VARCHAR(15) NOT NULL,
    Profit_percent NUMERIC(4,2),
    Unit_measure VARCHAR(10),
    Qty_on_hand NUMERIC(8) check (Qty_on_hand > 0),
    Recorder_level NUMERIC(8),
    Sell_price NUMERIC(8,2),
    Cost_price NUMERIC(8,2)
);
select * from Product_master
```

Data output Messages Notifications

product_no [PK] character varying (6)	description character varying (15)	profit_percent numeric (4,2)	unit_measure character varying (10)	qty_on_hand numeric (8)	recorder_level numeric (8)	sell_price numeric (8,2)	cost_price numeric (8,2)

Q2

PostgreSQL

```sql
INSERT INTO client_master
VALUES
('C001', 'John', '123 Main', 'Mumbai', 400091, 'Maharashtra', 20000.00),
('C002', 'Doe', '456 Elm', 'Kolkata', 300023, 'West Bengal', 50000.00),
('C003', 'Alfred', '789 Oak', 'Pune', 720001, 'Maharashtra', 2000.00),
('C004', 'James', '321 Map', 'Chennai', 498880, 'Tamil Nadu', 100000.00),
('C005', 'Dalton', '567 Pine', 'Mumbai', 560000, 'Karnataka', 70000.00);

SELECT * FROM client_master
```

client_master - TABLE

client_no	name	address	city	pincode	state	bal_due
C001	John	123 Main	Mumbai	400091	Maharashtra	20000.00
C002	Doe	456 Elm	Kolkata	300023	West Bengal	50000.00
C003	Alfred	789 Oak	Pune	720001	Maharashtra	2000.00
C004	James	321 Map	Chennai	498880	Tamil Nadu	100000.00
C005	Dalton	567 Pine	Mumbai	560000	Karnataka	70000.00

```sql
INSERT INTO product_master
VALUES
('P001', 'Laptop', 20.00, '2kg', 200, 2, 20000.00, 15000.00),
('P002', 'Hard disk', 10.00, '500g', 80, 2, 1500.00, 500.00),
('P003', 'Processor', 70.00, '3kg', 150, 2, 70000.00, 3500.00),
('P004', 'Keypad', 10.00, '500g', 70, 2, 2000.00, 100.00),
('P005', 'Printer', 30.00, '1.5kg', 300, 2, 10000.00, 1500.00);

SELECT * FROM product_master
```

product_...	description	profit_perc...	unit_measure	qty_on_hand	recorder_le...	sell_price	cost_price
P001	Laptop	20.00	2kg	200	2	20000.00	15000.00
P002	Hard disk	10.00	500g	80	2	1500.00	500.00
P003	Processor	70.00	3kg	150	2	70000.00	3500.00
P004	Keypad	10.00	500g	70	2	2000.00	100.00
P005	Printer	30.00	1.5kg	300	2	10000.00	1500.00

PostgreSQL

```sql
SELECT NAME FROM client_master
```

name
John
Doe
Alfred
James
Dalton

4

PostgreSQL

```sql
INSERT INTO client_master
VALUES
('C001', 'John', '123 Main', 'Mumbai', 400091, 'Maharashtra', 20000.00),
('C002', 'Doe', '456 Elm', 'Kolkata', 300023, 'West Bengal', 50000.00),
('C003', 'Alfred', '789 Oak', 'Pune', 720001, 'Maharashtra', 2000.00),
('C004', 'James', '321 Map', 'Chennai', 498880, 'Tamil Nadu', 100000.00),
('C005', 'Dalton', '567 Pine', 'Mumbai', 560000, 'Karnataka', 70000.00);

SELECT * FROM client_master
```

client_master - TABLE

client_no	name	address	city	pincode	state	bal_due
C001	John	123 Main	Mumbai	400091	Maharashtra	20000.00
C002	Doe	456 Elm	Kolkata	300023	West Bengal	50000.00
C003	Alfred	789 Oak	Pune	720001	Maharashtra	2000.00
C004	James	321 Map	Chennai	498880	Tamil Nadu	100000.00
C005	Dalton	567 Pine	Mumbai	560000	Karnataka	70000.00

5

PostgreSQL

```
1 SELECT NAME, city FROM client_master
```

name	city
John	Mumbai
Doe	Kolkata
Alfred	Pune
James	Chennai
Dalton	Mumbai

6

PostgreSQL

```
1 SELECT description FROM product_master
```

description
Laptop
Hard disk
Processor
Keypad
Printer

7

PostgreSQL

```
1 SELECT * FROM client_master WHERE city = 'Mumbai';
2
```

client_no	name	address	city	pincode	state	bal_due
C001	John	123 Main	Mumbai	400091	Maharashtra	20000.00
C005	Dalton	567 Pine	Mumbai	560000	Karnataka	70000.00

8

```
1 UPDATE client_master SET city = 'Mumbai' WHERE client_no = 'C001';
```

History

Syntax | History

PostgreSQL

UPDATE client_master SET city = 'Mumbai

20:54:09

9

```
1 UPDATE client_master SET bal_due = 1000 WHERE client_no = 'C005';
2 SELECT * FROM client_master
```

client_no	name	address	city	pincode	state	bal_due
C002	Doe	456 Elm	Kolkata	300023	West Bengal	50000.00
C003	Alfred	789 Oak	Pune	720001	Maharashtra	2000.00
C004	James	321 Map	Chennai	498880	Tamil Nadu	100000.00
C001	John	123 Main	Mumbai	400091	Maharashtra	20000.00
C005	Dalton	567 Pine	Mumbai	560000	Karnataka	1000.00

10

```sql
UPDATE product_master SET Cost_price = 3000 WHERE description = 'Hard disk';
SELECT * FROM product_master
```

product_...	description	profit_perc...	unit_measure	qty_on_hand	recorder_le...	sell_price	cost_price
P001	Laptop	20.00	2kg	200	2	20000.00	15000.00
P003	Processor	70.00	3kg	150	2	70000.00	3500.00
P004	Keypad	10.00	500g	70	2	2000.00	100.00
P005	Printer	30.00	1.5kg	300	2	10000.00	1500.00
P002	Hard disk	10.00	500g	80	2	1500.00	3000.00

11

```sql
DELETE FROM product_master WHERE qty_on_hand < 100;
SELECT * FROM product_master
```

product_...	description	profit_perc...	unit_measure	qty_on_hand	recorder_le...	sell_price	cost_price
P001	Laptop	20.00	2kg	200	2	20000.00	15000.00
P003	Processor	70.00	3kg	150	2	70000.00	3500.00
P005	Printer	30.00	1.5kg	300	2	10000.00	1500.00

12

client_no	name	address	city	pincode	state	bal_due
C002	Doe	456 Elm	Kolkata	300023	West Bengal	50000.00
C003	Alfred	789 Oak	Pune	720001	Maharashtra	2000.00
C001	John	123 Main	Mumbai	400091	Maharashtra	20000.00
C005	Dalton	567 Pine	Mumbai	560000	Karnataka	1000.00

Customer
- customer id
- name
- number
- address
- credit card no

Cart contains

pays
- customer id
- transcaction id

Cart contains
- Transaction id
- date and time
- Total cost
- quantity
- item id

items distributors
- item id
- image
- name
- price
- description

Shipper
- Contact no
- sname

1. Integer Types:
    - MySQL:
        - TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT
    - PostgreSQL:
        - SMALLINT, INTEGER, BIGINT
2. Decimal/Floating-Point Types:
    - MySQL:
        - DECIMAL, FLOAT, DOUBLE
    - PostgreSQL:
        - DECIMAL, NUMERIC, REAL, DOUBLE PRECISION
3. String/Character Types:
    - MySQL:
        - CHAR, VARCHAR, TEXT
    - PostgreSQL:
        - CHAR, VARCHAR, TEXT
4. Date and Time Types:
    - MySQL:
        - DATE, TIME, DATETIME, TIMESTAMP
    - PostgreSQL:
        - DATE, TIME, TIMESTAMP, INTERVAL
5. Boolean Type:
    - MySQL:
        - BOOLEAN
    - PostgreSQL:
        - BOOLEAN

DELETE:

The DELETE statement is used to remove specific rows from a table based on a condition specified in the WHERE clause.

It allows more flexibility as you can delete specific rows that meet certain criteria.

TRUNCATE:

The TRUNCATE statement is used to remove all rows from a table.

It removes all rows without considering any conditions. It effectively deletes all data from the table.