**SQL ASSIGNMENT**

1. **What is SQL?**

* SQL Stands for structured query language
* SQL lets you access and manipulate databases
* SQL can execute queries against a database
* SQL can retrieve data from a database
* SQL can insert a records in a database
* SQL can update a records in a database
* SQL can delete a records in a database
* SQL can create a new databases
* SQL can create stored procedures in a database
* SQL can create views in a database
* SQL can set permissions on tables, procedures and views
* Major commands in SQL are CREATE, ALTER, DROP, TRANCATE, RENAME, SELECT, INSERT, UPDATE, DELETE, REVOKE, GRANT , COMMIT, ROLLBACK AND SAVEPOINT.

.

1. Explain Commit, Rollback and Savepoint.

These three commands are used to control transactions.

**COMMIT :**

COMMIT is used to save the changes made to the table permanently.

**Syntax:**

COMMIT;

**ROLLBACK:**

* ROLL BACK is used to get back to the previous satus of the table similar to the UNDO operation.
* Table can be roll back only if table is temporary. If you committed your changes, it cannot be rollback.

**Syntax:**

ROLLBACK;

**SAVEPOINT:**

* SAVE POINT IS USED ALONG with the roll back command.
* It is used to mark a transaction in a table, a transaction can be named using this command**.**

**Syntax:**

SAVEPOINT SAVEPOINT\_NAME;

**Syntax for rolling back to a SAVEPOINT :**

ROLLBACK TO SAVEPOINT\_NAME;

1. **Explain DDL commands with example?**

* DDL stands for data definition language.
* DDL commands are

Create, alter, drop, truncate, rename

**Create :**

Create is used to create database or its objects(like table, index, function, views, store procedure and triggers).

**Example :**

Create table student(s\_id number(10), s\_name varchar2(20));

**Drop:**

Drop is used to delete objects from the database.

**Syntax :**

Drop table <table name>;

**Example:**

Drop table student;

**Truncate :**

Truncate is used to remove all records from a table, including all spaces allocated for the records are removed.

**Syntax** :

Truncate table <table name>;

**Example :**

Truncate table student;

**Alter :**

Alter is used too alter the structure of the database.

**Syntax :**

Alter table <table name> add <new\_column\_name>;

**Example :**

Alter table student add (age number(2));

**Rename:**

Rename is used rename an object existing in the database.

**Syntax :**

Rename <old\_name> to <new\_name>;

**Example** :

Rename student to scholars;

1. **Explain the difference between TRUNCATE, DELETE AND DROP.**

|  |  |  |
| --- | --- | --- |
| **TRUNCATE** | **DELETE** | **DROP** |
| TRUNCATE command is a data definition language operation . | DELETE command is a data manipulation language operation. | DROP command is a data definition language operation. |
| It is used to remove all records from a table. | It is used to delete existing records from an existing table. | It is used to delete existing database objects. |
| It deletes all the records from an existing table but not the table itself. | We can delete a single record or multiple records depending on the condition specified in the query | It can be used to delete databases, tables, views, triggers etc. |
| Syntax :  Truncate table <table\_name>; | Syntax :  Delete from <table\_name> where <condition>; | Syntax :  Drop table <table\_name>; |
| Example :  Truncate table student; | Example :  Delete from student where s\_id=1; | Example :  Drop table student; |
| Truncate statement is equivalent to delete operation without a where clause. The truncate command removes the records from a table without scanning it. This is why it is faster than delete statement . | The conditions specified in the where clause of the delete statement. If we omit the where clause then all of the records will be deleted and the table will be empty. | Drop command removes the table definition and all the data, indexes , triggers, constraints and the permission specifications for that table. |
| Truncate table statement is a DDL command so it can not be roll back. | Delete is a DML command so it can be rolled back. | Drop table statement is a DDL command so it can not be roll back. |

1. **Write a sql statement to retrieve those client from customer table who live in delhi, Bhwneshwar and Mumbai.**

**Sql statements :**

create table customer(c\_id number(10),c\_name varchar2(20),c\_city varchar2(20));

insert into customer values(1,'hanisha','chennai');

insert into customer values(2,'anusha','madhurai');

insert into customer values(3,'manisha','bhuwaheswar');

insert into customer values(4,'manasvi','mumbai');

insert into customer values(5,'jyothi','bhwaneswar');

insert into customer values(6,'pavaneswari','mumbai');

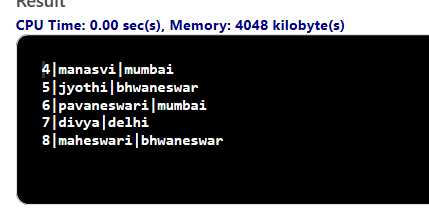
insert into customer values(7,'divya','delhi');

insert into customer values(8,'maheswari','bhwaneswar');

**// Write a sql statement to retrieve those client from customer table who live in delhi, Bhwneshwar and Mumbai.**

select \* from customer where c\_city = 'bhwaneswar' or c\_city = 'mumbai' or c\_city ='delhi';

**output :**

****

1. **Explain Where clause with example and how it’s different from having ?**

**WHERE clause :**

* + 1. The where clause is used to filter records and it is used to extract only those records that fulfill specified condition.
    2. The where clause is used in select, update, delete etc.

**Syntax :**

Select column from table\_name where <condition>;

**Example :**

Select age from student where s\_id=1;

**How it’s different from having:**

* The main differences between WHERE and HAVING clause comes when used together with group by clause, in that case where is used to filter rows before grouping and having is used to exclude records after grouping.
* Another key differences between where and having clause is that where will be use index and having will not.

1. **Explain ACID properties with suitable example?**

A transaction is a single logical unit of work which accesses and possibly modifies the content of a database.transaction access data using read and write operations.in oder to maintain consistency in a database,before and after the transaction, certain properties are followed. These are called ACID properties.

**Atomicity :**

By this, we mean that either the entire transaction takes place at once or does not happen at all.

It involves the two operations :

**Abort** :

If a transaction aborts, changes made to database are not visible .

**Commit :**

If a transaction commits, changes made are visible .

* + - Atomicity is also known as the “all or nothing rule”.

**Example :**

Consider the following transactions T consisting of T1 and T2 : transfer of 100 from account x to y.

Before x=400 and y=200;

Transaction T

T1 T2

Read(x); Read(Y);

X=x-100; Y=Y+100;

Write(x); Write(Y);

After x=300 After Y=300

the amount has been deducted from x and added to y.

**Consistency :**

This means that integrity constraints must be maintained so that database is consistent before and after the transaction. It refers to the correctness of a database .

**Example :**

The total amount before and after the transaction must be maintained.

From above example,

Total before T occurs = 400+200 =600

Total after T occurs = 300+300 =600

Therefore, database is consistent

**Isolation** :

This property ensures that multiple transactions can occur concurrently without leading to he inconsistency of database state. Transaction occur independently without interferences. Changes occurring in a particular transaction will not be visible to any other transaction until that particular change in that transaction is written to a memory or has been committed. this property ensures that the execution of a transaction concurrently will result in a state that is equivalent to a state achieved these were executed serially in same order.

**Durability :**

This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs. These updates now become permanent and are stored in non-violatile memory. the effects of the transaction. Thus, Never lost.

1. **Create the above mentioned tables and populate the tables accordingly.** 
   1. **CUSTOMER (CID, CNAME)**
   2. **BRANCH (BCODE, BNAME,BADDRESS). (BADDRESS can be Sadar, Civil Lines, Nawab Ganj and Sipri)**
   3. **ACCOUNT (ANO, ATYPE, BALANCE, CID, BCODE). An account can be a savings account or a current account.**
   4. **TRANSACTION (TID, ANO, TTYPE, TDATE, TAMOUNT) TTYPE CAN BE - Deposit OR Withdrawal**
   5. **BALANCE(cid,amount)**

CREATE TABLE CUSTOMER(CID NUMBER(20)PRIMARY KEY,CNAME VARCHAR(20));

CREATE TABLE BRANCH(BCODE NUMBER(20)PRIMARY KEY,BNAME VARCHAR(20),BADDERESES VARCHAR(20));

CREATE TABLE ACCOUNT(ANO NUMBER(20)PRIMARY KEY,ATYPE VARCHAR(20),BALANCE NUMBER(20),CID NUMBER(20) REFERENCES CUSTOMER(CID),BCODE NUMBER(20) REFERENCES BRANCH(BCODE));

CREATE TABLE TRANSACTION1(TID NUMBER(20)PRIMARY KEY,ANO NUMBER(20) REFERENCES ACCOUNT(ANO),TTYPE VARCHAR(20),TDATE DATE,TAMOUNT NUMBER(20));

CREATE TABLE BALANCE(CID NUMBER(20) REFERENCES CUSTOMER(CID),AMOUNT NUMBER(20));

INSERT INTO CUSTOMER VALUES(1, 'NISHA');

INSERT INTO CUSTOMER VALUES(2,'HASINI');

INSERT INTO CUSTOMER VALUES(3,'HANI');

INSERT INTO CUSTOMER VALUES(4,'ANISHA');

INSERT INTO CUSTOMER VALUES(53,'ANUSHA');

INSERT INTO CUSTOMER VALUES(6,'MANISHA');

INSERT INTO BRANCH VALUES(100,'VIJAYAWADA','NAWAB GAGJ');

INSERT INTO BRANCH VALUES(101,'PODILI','CIVIL LINES');

INSERT INTO BRANCH VALUES(102,'CHIRALA','SIPRI');

INSERT INTO BRANCH VALUES(103,'MACHAVARAM','SADAR');

INSERT INTO ACCOUNT VALUES(1000,'SAVINGS ACCOUNT',500,1,100);

INSERT INTO ACCOUNT VALUES(1001,'SAVINGS ACCOUNT',5500,2,101);

INSERT INTO ACCOUNT VALUES(1002,'SAVINGS ACCOUNT', 55500, 3, 102);

INSERT INTO ACCOUNT VALUES(1003, 'CURRENT ACCOUNT', 500000, 4, 103);

INSERT INTO ACCOUNT VALUES(1004,'CURRENT ACCOUNT',50,53,100);

INSERT INTO ACCOUNT VALUES(1005, 'CURRENT ACCOUNT', 5200, 6, 101);

INSERT INTO TRANSACTION1 VALUES(10000, 1000, 'SAVINGS ACCOUNT', '22-02-2020', 100);

INSERT INTO TRANSACTION1 VALUES(10001, 1001, 'SAVINGS ACCOUNT', '02-022020', 1000);

INSERT INTO TRANSACTION1 VALUES(10002, 1002, 'SAVINGS ACCOUNT', '22-03-2020', 5000);

INSERT INTO TRANSACTION1 VALUES(10003, 1003,' SAVINGS ACCOUNT', '22-04-020', 100000);

INSERT INTO TRANSACTION1 VALUES(10004, 1004, 'SAVINGS ACCOUNT', '18-03-2020', 200);

INSERT INTO TRANSACTION1 VALUES(10005,1005,'SAVINGS ACCOUNT','10-02-2020',100);

INSERT INTO BALANCE VALUES(1,400);

INSERT INTO BALANCE VALUES(2,4500);

INSERT INTO BALANCE VALUES(3,50500);

INSERT INTO BALANCE VALUES(4,400000);

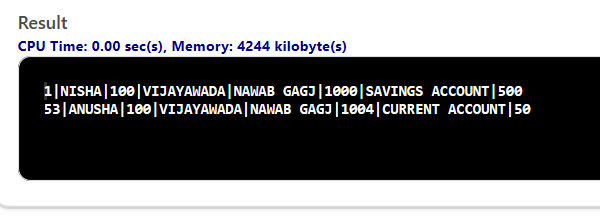
INSERT INTO BALANCE VALUES(5,30);

INSERT INTO BALANCE VALUES(6,5100);

**9.Write a SQL query to get the details of all customers whose account balance is less than 1000.**

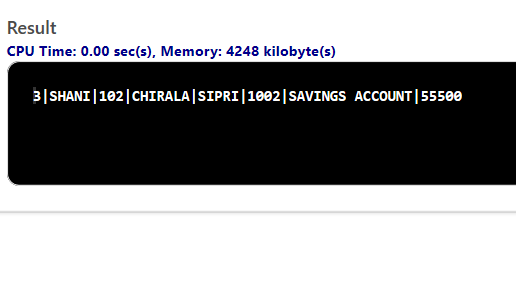
SELECT C.CID,C.CNAME, B.BCODE, B.BNAME, B.BADDERESES, A.ANO, A.ATYPE, A.BALANCE FROM CUSTOMER C, BRANCH B, ACCOUNT A WHERE A.BALANCE<1000 AND C.CID=A.CID AND B.BCODE=A.BCODE;

**OUTPUT :**

****

**10.Write a SQL query to get those records whose customers name start with SH.**

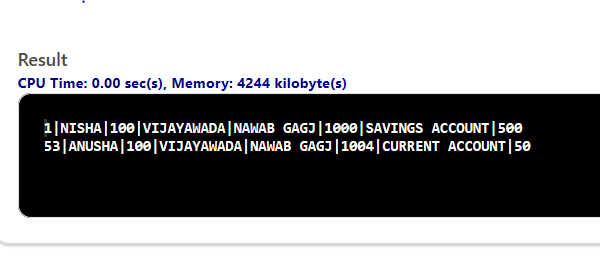
SELECT C.CID, C.CNAME, B.BCODE, B.BNAME, B.BADDERESES, A.ANO, A.ATYPE, A.BALANCE FROM CUSTOMER C, BRANCH B, ACCOUNT A WHERE C.CNAME LIKE 'SH%' AND C.CID=A.CID AND B.BCODE=A.BCODE;

****

**11.Write A SQL QUERY TO GET THE DETAILS OF THOSE CUSTOMERS WHO ARE HAVING MORE THAN 50000 Rs. Balance in their account .**

SELECT C.CID, C.CNAME, B.BCODE, B.BNAME, B.BADDERESES, A.ANO, A.ATYPE, A.BALANCE FROM CUSTOMER C, BRANCH B, ACCOUNT A WHERE A.BALANCE>50000 AND C.CID=A.CID AND B.BCODE=A.BCODE;

**OUTPUT :**

****

**12.Delete all the record from transAction1 table in one go but not table.**

DELETE FROM TRANSACTION1;

SELECT \* FROM TRANSACTION1;

**OUTPUT :**

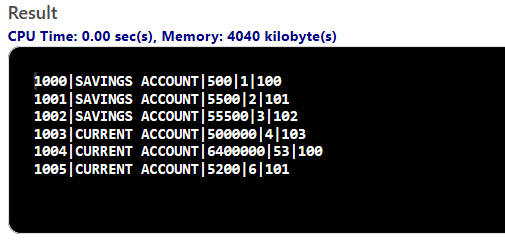
****

**13.Update balance of Rs.6400000 whose customer id  is 53 .**

UPDATE ACCOUNT SET BALANCE=6400000 WHERE CID=53;

SELECT \* FROM ACCOUNT;

**OUTPUT :**

****

**14.Change the name of CNAME column to C\_NAME.**

ALTER TABLE CUSTOMER RENAME CNAME TO C\_NAME;

**15. Add one more column to BALANCE TABLE name account .**

ALTER TABLE BALANCE ADD ACCOUNT VARCHAR(20);