SE Comp - B	Roll number : 8942
Experiment no. : 10	Date of Implementation : 13-05-2021

Aim: Simple Transaction implementation

Tool Used : MySql/PostgreSQL
https://www.javatpoint.com/mysql-transaction

Related Course outcome : At the end of the course, Students will be able to Use and Apply the concept of transaction, concurrency and recovery

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)**
Implementation of concepts (3)	**N/A**	**< 80% complete (1-2)**	**100% complete (3)**
Originality • **Extent of plagiarism(2)**	**Copied it from someone else(0)**	**At least few parts of it have been done without copying(1)**	**Experiment 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)**

Rubrics for assessment of Experiment:

Assessment Marks :

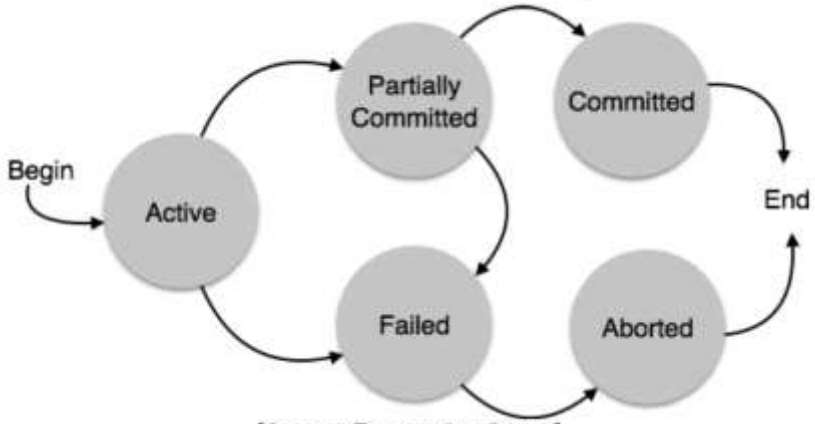
Timeliness	
Completeness and neatness	
Originality	
Knowledge	
Total	

Total : (Out of 10)

Teacher's Sign :

EXPERIMENT 10	***Transaction concept***
Aim	To implement Simple Transaction concept
Tools	Mysql/PostgreSQL

Theory	A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.
	Transactions are a fundamental concept of all database systems. The essential point of a transaction is that it bundles multiple steps into a single, all-or-nothing operation. The intermediate states between the steps are not visible to other concurrent transactions, and if some failure occurs that prevents the transaction from completing, then none of the steps affect the database at all.
	Properties of Transactions
	Transactions have the following four standard properties, usually referred to by the acronym ACID −
	• **Atomicity** − Ensures that all operations within the work unit are completed successfully; otherwise, the transaction is aborted at the point of failure and previous operations are rolled back to their former state.
	• **Consistency** − Ensures that the database properly changes states upon a successfully committed transaction.
	• **Isolation** − Enables transactions to operate independently of and transparent to each other.
	• **Durability** − Ensures that the result or effect of a committed transaction persists in case of a system failure.
	
	In PostgreSQL, a transaction is set up by surrounding the SQL commands of the transaction with BEGIN and COMMIT commands. So our banking transaction would actually look like:
	BEGIN;
	UPDATE accounts SET balance = balance - 100.00
	  WHERE name = 'Alice';
	-- etc etc
	COMMIT;
	End;

Theory	**State Diagram :** A transaction in a database can be in one of the following states:  [*Image: Transaction States*] For example, consider a bank database that contains balances for various customer accounts, as well as total deposit balances for branches. Suppose that we want to record a payment of $100.00 from Alice's account to Bob's account. BEGIN; --sql SAVEPOINT my_savepoint; UPDATE accounts SET balance = balance - 100.00 WHERE name = 'Alice'; UPDATE accounts SET balance = balance + 100.00 WHERE name = 'Bob'; ROLLBACK TO my_savepoint; or commit; --UPDATE accounts SET balance = balance + 100.00 WHERE name = 'Wally'; COMMIT;

Theory	# Transaction Control (TCL) The following commands are used to control transactions − • **BEGIN TRANSACTION** − To start a transaction. • **COMMIT** − To save the changes, alternatively you can use **END TRANSACTION** command. • **ROLLBACK** − To rollback the changes. Transactional control commands are only used with the DML commands INSERT, UPDATE and DELETE only. They cannot be used while creating tables or dropping them because these operations are automatically committed in the database. ## The BEGIN TRANSACTION Command Transactions can be started using BEGIN TRANSACTION or simply BEGIN command. Such transactions usually persist until the next COMMIT or ROLLBACK command is encountered. But a transaction will also ROLLBACK if the database is closed or if an error occurs. **The following is the simple syntax to start a transaction −** **BEGIN;** **or** **BEGIN TRANSACTION;** **The COMMIT Command** **The COMMIT command is the transactional command used to save changes invoked by a transaction to the database.** **The COMMIT command saves all transactions to the database since the last COMMIT or ROLLBACK command.** **The syntax for COMMIT command is as follows −** **COMMIT;** **or** **END TRANSACTION;**

Theory	**The ROLLBACK Command** The ROLLBACK command is the transactional command used to undo transactions that have not already been saved to the database. The ROLLBACK command can only be used to undo transactions since the last COMMIT or ROLLBACK command was issued. The syntax for ROLLBACK command is as follows − ROLLBACK;
Task	**Task1: Perform following task** create table student with column (id, name) start transaction; Insert following records (1, 'Amita') (2, 'Sheena') (3, 'Lavina') (4, 'Rex') (5, "Rahul') Update name of id 5 form 'Rahul' to 'Abhijit' Create a save point A; Insert new record (6, 'chris') Create a save point B; Insert new record (7, 'Bravo') Create a save point C; Display all rows of the students table (select * from students) Observe the output **Task 2: Rollback to save point B and observe the output** Perform task 2 and observe the output and explain the output **Task 3: Rollback to save point A and observe the output** Perform task 3 and observe the output and explain the output **Task 4: Now delete record of 'Rex', before delete create a save point , and rollback to this save point to undo this delete operation** Perform task 4 and observe the output and explain the output **Task 5: Now Perform commit** Perform task 5 and observe the output and explain the output
Links	https://www.studytonight.com/dbms/tcl-command.php https://www.splessons.com/lesson/mysql-tcl/ **https://www.tutorialspoint.com/sql/sql-transactions.htm**

Post Lab Questions:	1. Explain set transaction command in SQL 2. Explain how do you remove a savepoint (checkpoint) that you have created?

Task1: Perform following task

CODE and OUTPUT:

```
create table student(id int, name varchar(20));
start transaction;
insert into student values(1,'Amita');
insert into student values(2,'Sheena');
insert into student values(3,'Lavina');
insert into student values(4,'Rex');
insert into student values(5,'Rahul');
```

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Rahul

```
SET SQL_SAFE_UPDATES = 0;
UPDATE student SET name = 'Abhijit' WHERE id = '5';
```

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit

SAVEPOINT A;
INSERT INTO student VALUES(6, 'Chris');

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit
6	Chris

SAVEPOINT B;
INSERT INTO student VALUES (7, 'Bravo');

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit
6	Chris
7	Bravo

SAVEPOINT C;
SELECT * FROM student;

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit
6	Chris
7	Bravo

EXPLAINATION:

Here, Firstly we are creating a table student then using SET TRANSACTION placing a name on a transaction. Further inserting 5 values to the table. Then using update command we are trying to update the name to Abhijit for the id='5'. Later using the savepoint command we created a savepoint A. Using this command we can name the different states of our data in any table and then rollback to that state using the ROLLBACK command whenever required. After that we inserted into student for id='6', which was successfully executed. Then we created a savepoint B and inserted into student for id='7', which was successfully inserted to the table. Later we created another savepoint C and displayed all the values present in the student table. There were total 7 entries in the final output after the execution of task1.

Task 2: Rollback to save point B and observe the output
Perform task 2 and observe the output and explain the output

CODE:
rollback to B;

OUTPUT:

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit
6	Chris

EXPLAINATION:

The transaction is rolled back to savepoint B, so whatever operations performed after save point B will be rolled back.

So record of 'Bravo' that was inserted after save point B will be rolled back.

Task 3: Rollback to save point A and observe the output
Perform task 3 and observe the output and explain the output

CODE:
rollback to A;

OUTPUT:

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit

EXPLAINATION:

The transaction is rolled back to savepoint A, so whatever operations performed after save point A will be rolled back.

So After save point A we inserted record of ‘chris’ and ‘Bravo’ so this insert statements will be rolled back and hence table contains only 5 records.

Task 4: Now delete record of ‘Rex’, before delete create a save point , and rollback to this save point to undo this delete operation
Perform task 4 and observe the output and explain the output

CODE:
savepoint DD;
SET SQL_SAFE_UPDATES = 0;
delete from student where id =4;
select * from student;

OUTPUT:

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
5	Abhijit

EXPLAINATION:

Here the record of Rex present at id='4' is successfully deleted using the delete command.

- **Now we want to undo this delete operation, so perform rollback to check point DD**

CODE:

```
rollback to DD;
select * from student;
```

OUTPUT:

Result Grid

id	name
1	Amita
2	Sheena
3	Lavina
4	Rex
5	Abhijit

EXPLAINATION:

Here we are performing rollback to savepoint DD, as we want to undo the delete operation that we performed previously.

Task 5: Now Perform commit
Perform task 5 and observe the output and explain the output

CODE:
Commit;

OUTPUT:

EXPLAINATION:

Now these 5 records have been committed so if you try to rollback now, you would get an error.

<u>POSTLAB QUESTIONS</u>:

1. Explain set transaction command in SQL
Ans:

- The SET TRANSACTION command can be used to initiate a database transaction. This command is used to specify characteristics for the transaction that follows. For example, you can specify a transaction to be read only or read write.

- The syntax for a SET TRANSACTION command is as follows.

  SET TRANSACTION [READ WRITE | READ ONLY];

- Use the SET TRANSACTION statement to establish the current transaction as read-only or read/write, establish its isolation level, or assign it to a specified rollback segment.

- The operations performed by a SET TRANSACTION statement affect only your current transaction, not other users or other transactions. Your transaction ends whenever you issue a COMMIT or ROLLBACK statement. Oracle Database implicitly commits the current transaction before and after executing a data definition language (DDL) statement.

2. Explain how do you remove a savepoint (checkpoint) that you have created?

Ans:

- The RELEASE SAVEPOINT statement removes the named savepoint from the set of savepoints of the current transaction. No commit or rollback occurs. It is an error if the savepoint does not exist.

- All savepoints of the current transaction are deleted if you execute a COMMIT, or a ROLLBACK that does not name a savepoint.

- A new savepoint level is created when a stored function is invoked or a trigger is activated. The savepoints on previous levels become unavailable and thus do not conflict with savepoints on the new level. When the function or trigger terminates, any savepoints it created are released and the previous savepoint level is restored.

- Note that it is possible to also manually delete a savepoint via regular file system operations without affecting other savepoints or checkpoints (recall that each savepoint is self-contained)