1. Start two connections to your database server (note that both connections can be made with the same user i.e. root). On the first instance/connection, execute the following. Red text will indicate the first instance; blue text will indicate the second.
2. Write a query that calculates the number of records in the example1 table and provide this query. (1 mark)

**SELECT COUNT(\*) FROM example1;**

****

1. Execute the query from 1. How many records are stored in the example1 table? (1 mark)

**100 records**

1. Execute the query below.

start transaction;

insert into example1(value1) values(5);

insert into example1(value1) values(25);

1. Execute the insert statements from 1 again. How many records are stored in the example1 table? (1 mark)

**102 records**

****

1. Switch to the second instance/connection. Set the default database to transtest.

USE transtest;

1. How many records are stored in the example1 table? Why? Which letter of ACID does it demonstrate? (1 mark)

**100 records. This demonstrates the I for isolation in ACID. Multiple transactions that occur simultaneously should not impact each other’s execution.**

1. Switch to the first instance/connection and ensure safe updates is disabled.

SET SQL\_SAFE\_UPDATES=0;

*Or go to Edit --> Preferences --> SQL Editor and uncheck “Safe Updates” at the bottom of the window. If you did not have this setting set, you will have to restart your connection to the first instance for this change to take effect*.

1. Run the following command:

DELETE FROM example1 WHERE value1 <50;

1. How many records are stored in the example1 table in the first instance? (1 mark)

**There are 92 records left.**

1. Commit the transaction from the first instance then switch to the second instance/connection.
2. How many records are stored in the example1 table in the second instance? Explain. (1 mark)

**There are also 92 records in the second instance. The COMMIT statement saves the changes made to the table/database and makes all the changes visible to the other users.**

1. Switch back to the first instance/connection and run the following:

start transaction;

insert into example1(value1) values(900);

insert into example1(value1) values(901);

insert into example1(id,value1) values(5,902);

commit;

*Note that the last insert generates a primary key conflict.*

*Note 2.  MySQL is behaving badly at this point.  Most database servers would do an automatic rollback*

1. How many records are stored in the example1 table in the first instance? (1 mark)

**There are 94 records in the table.**

1. Switch to the second instance/connection.
2. What is the number of records in the example1 table in the second instance? (1 mark)

**There are 92 records in the second instance.**

1. Which letter of ACID does it demonstrate? (1 mark)

**It demonstrates the A for atomicity in ACID. The last INSERT statement failed, as a result the entire transaction also failed.**

1. Switch back to the first instance and run  
   commit;
2. How many records are stored in the example1 table in the first instance? (1 mark)

**There are now 94 records.**

1. Switch to the second instance/connection.
2. How many records are stored in the example1 table in the second instance? Explain. (1 mark)

**There are also 94 records; The first transaction failed and was not successfully committed. However, the table was still changed, so the second COMMIT saved the changes made to the table.**

1. Switch to the first instance/connection. Execute the entire transaction below:

start transaction;

insert into example1(value1) values(701);

insert into example1(value1) values(703);

insert into example1(value1) values(704);

rollback;

1. How many records are stored in the example1 table in the first instance? (1 mark)

**There are still 94 records.**

1. Switch to the second instance/connection.
2. How many records are stored in the example1 table in the second instance? Explain. (1 mark)

**Also 94 records. The ROLLBACK statement undid the previous transactions that have not been committed yet.**

1. Switch to the first instance/connection and run:

start transaction;

insert into example1(value1) values(800);

savepoint i15;

insert into example1(value1) values(801);

savepoint i16;

insert into example1(value1) values(802);

savepoint i17;

rollback to savepoint i15;

1. How many records are stored in the example1 table in the first instance? (1 mark)

**There are 95 records.**

1. Switch to the second instance/connection.
2. How many records are stored in the example1 table in the second instance? (1 mark)

**There are 94 records.**

1. Switch to the first instance/connection and runt he following:

commit;

1. How many records are stored in the example1 table in the first instance? (1 mark)

**There are 95 records.**

1. Switch to the second instance/connection.
2. How many records are stored in the example1 table in the second instance? Explain (1 mark)

**There are also 95 records. The ROLLBACK statement undid transactions i16 and i17. As a result, only the changes made in SAVEPOINT i15 were committed.**