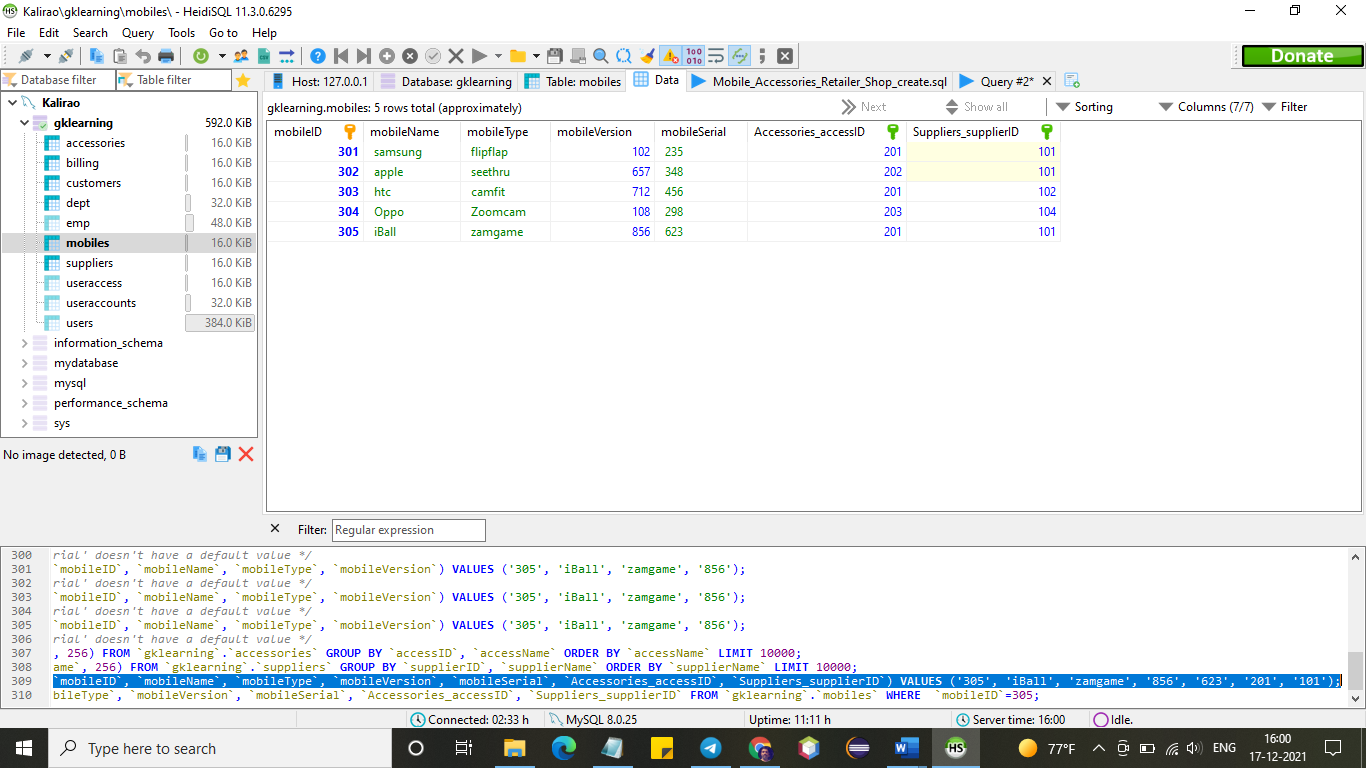
Transactions

ACID is an acronym that describes four properties of a robust database system: atomicity, consistency, isolation, and durability. These features are scoped to a transaction, which is a unit of work that the programmer can define. A transaction can combine one or more database operations.

**INSERT** **INTO** `gklearning`.`mobiles` (`mobileID`, `mobileName`, `mobileType`, `mobileVersion`, `mobileSerial`, `Accessories\_accessID`, `Suppliers\_supplierID`) **VALUES** ('305', 'iBall', 'zamgame', '856', '623', '201', '101');



Look at the above insert statement it describes certain amount of values are going to be placed into database in a row.

With atomicity, the statement is treated as a single unit, and thanks to consistency (the C in ACID) there are only two possible outcomes: either they all change the database or none of them do. This is important in situations where transfers between tables could result in disaster if the server were to go down after a DELETE statement but before the corresponding INSERT statement.

Like-wise with consistency, your command never leaves database in a half-finished state. If one part of this data transfer during insert/delete/update fails, all of the pending changes are rolled back, leaving the database as it was before you initiated the above statement.

Isolation is the activity that keep separates this insertion activity from other set of statements until it was finished.

A standard existence does not automatically makes make all implementer players adherent.

A RDBMS is ACID compliant if it provided ACID behavior in any way. Not every DMBS is ACID compliant (MySQL was NOT, until recently).

ACID is the ideal world. Isolation Levels are the real world.