Best Practices for InnoDB tables

1. Specifying a primary key for every table using the most frequently queried column or columns, or an auto-increment value if there is not obvious primary key.
2. Using joins wherever data is pulled from multiple table based on identical ID values from those tables. For fast join performance, define foreign keys on the join columns, and declare those columns with the same data type in each table. Adding foreign keys ensures that referenced columns are indexed, which can improve performance. Foreign keys also propagate deletes or updates to all affected tables, and prevent insertion of data in a child table if the corresponding IDs are not present in the parent table.
3. Turning off autocommit. Committing hundreds of times a second puts a cap on performance (limited by the write speed of your storage device).
4. Grouping sets of related DML operations into transactions, by bracketing them with START TRANSACTION and COMMIT statements. While you don't want to commit too often, you also don't want to issue huge batches of INSERT, UPDATE, or DELETE statements that run for hours without committing.
5. Not using LOCK TABLES statements. InnoDB can handle multiple sessions all reading and writing to the same table at once, without sacrificing reliability or high performance. To get exclusive write access to a set of rows, use the SELECT ... FOR UPDATE syntax to lock just the rows you intend to update.
6. Enabling the innodb file per table option to put the data and indexes for individual tables into separate files, instead of in a single giant system tablespace. This setting is required to use some of the other features, such as table compression and fast truncation.
The innodb file per table option is enabled by default as os MySQL 5.6.6.
7. Evaluating whether your data and access patterns benefit from the InnoDB table compression feature (ROW_FORMAT=COMPRESSED) on the CREATE TABLE statement. You can compress InnoDB tables without sacrificing read/write capability.
8. Running your server with the option --sql mode=NO ENGINE SUBSTITUTION to prevent tables being created with a different storage engine if there is an issue with the engine specified in the ENGINE= clause of CREATE TABLE.

参考 ： https://dev.mysql.com/doc/refman/5.7/en/innodb-best-practices.html