MySQL Cheat Sheet

SHOW

Show the current environment variables of MySQL
mysql> SHOW VARIABLES;

Show what each thread is doing when you execute it.
mysql> SHOW PROCESSLIST; or
mysql> SHOW PROCESSLIST \G (vertical display) **or**
mysql> SHOW FULL PROCESSLIST (include whole SQL statements being run)

Show counters and stats of how often events occur:
mysql> SHOW STATUS
mysql> FLUSH STATUS (reset the counters and stats)
mysql> SHOW INNODB STATUS (Innodb is MySQLs storage engine, reads as textual report like Show Status)
mysql> SHOW ENGINE STATUS (Future replacement of SHOW INNODB STATUS)

STORAGE ENGINES

MySQL can use one of 3 broad types of storage engine. Summarised here:
MyISAM, Heap, Merge: Table Lock on insert/update/delete (write lock) or select (read lock – multiuser)
BDB: (Page locks, e.g. Locks pages of 8kb at a time, not the whole table)
InnoDB: (Performs read/write locking on a per-record basis, increased overhead due to extra details being invisibly stored, e.g. Creation id, deletion id.

Assign Variables In Queries

You can select details in a query then use that result as a piece of your next query rather than using PHP to do it, this example uses @id:

```
SELECT @id := id FROM Stock WHERE name = 'AMZN';
INSERT INTO StockPrice VALUES (@id, '2002-05-03', 20.50);
```

COMMIT

You can perform SQL statements on an "All-Or-Nothing" basis, where if one portion of the script fails, the whole batch will be rolled back. This is known as a transaction. I will use the SQL above as a basis of explaination:

```
BEGIN;
SELECT @id := id FROM Stock WHERE name = 'AMZN';
INSERT INTO StockPrice VALUES (@id, '2002-05-03', 20.50);
COMMIT;
```

But there are various transaction modes to choose from as they do different jobs:

Read Uncommitted:	Known as a dirty read, allows to read during the transaction, probably for debugging only. Generally never used in practice due to the high chance of seeing incorrect data.
Read Committed:	Will see the data before the transaction started if query is run during the process. After the process it will see the result of the whole transaction. This is the default action.
Repeatable Read:	If you base one query of a table on the result of an SQL statement on another table, then the basis table changes during the execution of your original statement, you have a result of your original statement that may be no longer valid. A repeatable read locks any records that are read during a transaction, not just ones that are written.
Serializability:	Orders transactions so phantoms cannot occur. Highest level of isolation and as a result has the greatest performance overhead. Most reliable.