首先看SqlSession的commit()他的一个接口的方法，所以去他的实现类找（ctrl+h），DefaultSqlSession类中查找他的commit无参数的方法：

**public void** commit() {  
 **this**.commit(**false**);  
}  
  
**public void** commit(**boolean** force) {  
 **try** {  
 **this**.executor.commit(**this**.isCommitOrRollbackRequired(force));  
 **this**.dirty = **false**;  
 } **catch** (Exception var6) {  
 **throw** ExceptionFactory.wrapException(**"Error committing transaction. Cause: "** + var6, var6);  
 } **finally** {  
 ErrorContext.instance().reset();  
 }  
  
}

commit()方法中调用了有参的commit(boolean force)方法，并且传递进去false参数。

dirty变成false在执行器的提交下面，说明数据不是脏的了。

executor.commit(this.isCommitOrRollbackRequired(force))执行器的提交，看里面的方法：

**private boolean** isCommitOrRollbackRequired(**boolean** force) {  
 **return this**.dirty || force;  
}

commit在方法参数里面调用了方法并拿到返回值，传进去的值为false。return回去的是true。

上面的this.dirty是在上回说到的增删改查都会底层调用update方法，里面改为true的。

上面的autoCommit则创建sqlSession的时候就早早的改为false，

OpenSession方法底层

接下来看上面的上面的executor.commit(XXX)方法，，他是Executor执行器接口的，找他的实现类BaseExecutor类的commit的带boolean参数的方法：

**public void** commit(**boolean** required) **throws** SQLException {  
 **if** (**this**.closed) {  
 **throw new** ExecutorException(**"Cannot commit, transaction is already closed"**);  
 } **else** {  
 **this**.clearLocalCache();  
 **this**.flushStatements();  
 **if** (required) {  
 **this**.transaction.commit();  
 }  
  
 }  
}

里面有一些不需要关注，不过简单提一嘴，clearLocalCache()清理缓存，flushStatements刷新参数

最终级的关注点就是因为if中的required是传进来的参数，上面已经解释过是true，所以他执行下一行代码就是transaction.commit()。

终结一句话：session.commit()最终调度到了事物的提交 ,this.transaction.commit()