# 数据库锁的基本原理

为了保证数据的完事性和一致性，数据库系统采用锁来实现事务的隔离性。各种大型数据库采用的锁基本理论是一致的，但在具体实现上各有差别。

从并发事务锁定的关系上看，可以分为共享锁定和独占锁定。从锁定的对象不同，一般可以分为表锁定和行锁定。

#### 锁

共享锁用于读取数据操作，它是非独占的，允许其他事务同时读取其锁定的资源，但不允许其他事务更新它。

独占锁也叫排他锁，适用于修改数据的场合。它所锁定的资源，其他事务不能读取也不能修改。

当一个事务访问某种数据库资源时，如果执行select语句，必须先获得共享锁，如果执行insert、update或delete语句，必须获得独占锁，这些锁用于锁定被操作的资源。

当第二个事务也要访问相同的资源时，如果执行select语句，也必须先获得共享锁，如果执行insert、update或delete语句，也必须获得独占锁。此时根据已经旋转在资源上的锁的类型，来决定第二个事务应该等待第一个事务解除对应资源的锁定，还是可以立刻获得锁。

|  |  |  |
| --- | --- | --- |
| **资源上已经放置的锁** | **第二个事务进行读操作** | **第二个事务进行更新操作** |
| **无** | 立即获得共享锁 | 立即获得独占锁 |
| **共享锁** | 立即获得共享锁 | 等待第一个事务解除共享锁 |
| **独占锁** | 等待第一个事务解除独占锁 | 等待第一个事务解除独占锁 |

###### 1　共享锁

1、加锁的条件：当一个事务执行select语句时，数据库系统会为这个事务分配一把共享锁，来锁定被查询的数据。

2、解锁的条件：在默认情况下，数据被读取后，数据库系统立即解除共享锁。例如，当一个事务执行查询“SELECT \* FROM accounts”语句时，数据库系统首先锁定第一行，读取之后，解除对第一行的锁定，然后锁定第二行。这样，在一个事务读操作过程中，允许其他事务同时更新accounts表中未锁定的行。

3、与其他锁的兼容性：如果数据资源上放置了共享锁，还能再放置共享锁和更新锁。

4、并发性能：具有良好的并发性能，当数据被放置共享锁后，还可以再放置共享锁或更新锁。所以并发性能很好。

###### 2　独占锁

1、加锁的条件：当一个事务执行insert、update或delete语句时，数据库系统会自动对SQL语句操纵的数据资源使用独占锁。如果该数据资源已经有其他锁（任何锁）存在时，就无法对其再放置独占锁了。

2、解锁的条件：独占锁需要等到事务结束才能被解除。

3、兼容性：独占锁不能和其他锁兼容，如果数据资源上已经加了独占锁，就不能再放置其他的锁了。同样，如果数据资源上已经放置了其他锁，那么也就不能再放置独占锁了。

4、并发性能：不用说了，最差。只允许一个事务访问锁定的数据，如果其他事务也需要访问该数据，就必须等待，起到前一个事务结束，解除了独占锁，其他事务才有机会访问该数据。

###### 3　更新锁

更新锁在的初始化阶段用来锁定可能要被修改的资源，这可以避免使用共享锁造成的死锁现象。例如，对于以下的update语句：

|  |
| --- |
| UPDATE accounts SET balance=900 WHERE id=1 |

更新操作需要分两步：

* 读取accounts表中id为1的记录。
* 执行更新操作。

如果在第一步使用共享锁，再第二步把锁升级为独占锁，就可能出现死锁现象。例如：两个事务都获取了同一数据资源的共享锁，然后都要把锁升级为独占锁，但需要等待另一个事务解除共享锁才能升级为独占锁，这就造成了死锁。

更新锁有如下特征：

* 加锁的条件：当一个事务执行update语句时，数据库系统会先为事务分配一把更新锁。
* 解锁的条件：当读取数据完毕，执行更新操作时，会把更新锁升级为独占锁。
* 与其他锁的兼容性：更新锁与共享锁是兼容的，也就是说，一个资源可以同时放置更新锁和共享锁，但是最多放置一把更新锁。这样，当多个事务更新相同的数据时，只有一个事务能获得更新锁，然后再把更新锁升级为独占锁，其他事务必须等到前一个事务结束后，才能获取得更新锁，这就避免了死锁。
* 并发性能：允许多个事务同时读锁定的资源，但不允许其他事务修改它。