redo日志记录的格式

- redo日志的记录格式与undo日志一样，唯一的区别是：在更新记录<T, X, V>中记载的是更新后的值

redo日志的记载规则

- R_1：在修改磁盘上的任何数据库元素X之前，要保证所有与 X 的这一修改有关的日志记录(包括更新记录<T,X,V> 和提交记录 <Commit T>) 都必须出现在磁盘上。

	T	t	M-A	M-B	D-A	D-B	redo日志
1					8	8	<Start T>
2	Read(A,t)	8	8		8	8	
3	t := t*2	16	8		8	8	
4	Write(A,t)	16	16		8	8	<T, A, 16>
5	Read(B,t)	8	16	8	8	8	
6	t := t*2	16	16	8	8	8	
7	Write(B,t)	16	16	16	8	8	<T, B, 16>
8							<Commit T>
9	Flush Log						
10	Output(A)		16	16	16	8	
11	Output(B)		16	16	16	16	

❑事实上，Output(A)和 Output(B)可能发生得更晚。

							redo日志
1							<Start T>
2	Re						
3	t						
4	Wr						<T, A, 16>
5	Re						
6	t						
7	Write(B,		16	16	8	8	<T, B, 16>
8							<Commit T>
9	Flush Log						
10	Output(A)		16	16	16	8	
11	Output(B)		16	16	16	16	

执行‘**Flash Log**’操作的目的是为了确保与事务**T**有关的日志记录（*包括后像日志和提交日志*）能够及时被写入磁盘，一方面是保证事务**T**的真正提交，另一方面也为今后的**Output**操作提供保证（规则$\mathbf{R_1}$）

❑事实上，Output(A)和 Output(B)可能发生得更晚。

- 使用redo日志的恢复过程

1. 确定所有已提交的事务；
2. 从日志文件的首部开始扫描日志，对遇到的每一条更新记录<T, X, V>：
   - 如果T是未提交事务，则继续扫描日志；
   - 如果T是已提交的事务，则为数据库元素X写入新值V。
3. 对每个未完成的事务T，在日志的尾部写入结束标志<Abort T>并刷新日志。

- 以图2为例来看使用redo日志的恢复过程。如果系统故障发生在：

1) 第9步之后
   - 与事务T有关的日志记录已经全部写到磁盘

2) 第8步和第9步之间
   - 事务T的提交记录<Commit T>已经被记入日志，但还不确定该日志是否已经被写入日志磁盘

3) 第8步之前
   - T是一个未完成的事务

情况1)：故障发生在第9步之后

	T	t	M-A	M-B	D-A	D-B	redo日志
1					8	8	<Start T>
							<T,A,16>
							<T,B,16>
8							<Commit T>
9	Flush Log						
	系统崩溃						

- 此时，由于与事务T有关的日志记录已经全部写到磁盘，因此恢复子系统认为T是一个已提交的事务。在恢复过程中会根据更新记录将A和B的新值写入数据库的磁盘中(有可能是冗余的写操作)

情况2)：故障发生在第8步与第9步之间

	T	t	M-A	M-B	D-A	D-B	redo日志
1					8	8	<Start T>
							<T,A,16>
							<T,B,16>
8							<Commit T>?
	系统崩溃						

- 如果<Commit T>已到达磁盘，T被认为是一个已提交的事务，恢复过程同情况 1)
- 如果<Commit T>尚未到达磁盘， T被认为是一个未结束的事务，则恢复过程同情况 3)

情况3)：故障发生在第8步之前

	T	t	M-A	M-B	D-A	D-B	redo日志
1					8	8	<Start T>
2							
3							
4							<T,A,16>
5							
6	t := t*2	16	16	8	8	8	
7	Write(B,t)	16	16	16	8	8	<T,B,16>
	在第8步之前的任何时候出现系统崩溃						

- **T是一个未完成的事务，由于T尚未修改*磁盘*上的任何数据库元素，因此在恢复过程中将对T的更新记录<T,X,V>不做任何处理，只是在日志的尾部写入一条记录<Abort T>**

- **redo日志与undo日志的主要区别**
  1) 恢复的目的不一样
  2) 提交记录<Commit T>写入日志的时间不一样
     - undo日志：在事务T的所有数据库磁盘修改操作(Output)结束后才能在日志中写入提交记录<Commit T>
     - redo日志：在提交记录<Commit T>被写入日志文件的磁盘后才能将事务T更新后的值写入数据库的磁盘中
  3) 在更新记录<T,X,V>中保存的值V不一样