事务的基本概念

事务处理是对各企业的事务型应用业务的数据处理（例如商品销、货物订购、货币交易）。通常，事务型应用业务逻辑称为事务，是由一系列的数据库增删改查操作组成，它是事务处理的最小单元，也是应用程序的基本逻辑单元。

事务

事务是用户定义的一个数据库操作系列，用于描述应用业务的逻辑。在关系数据库中，一个事务可以是一条SQL语句，也可以是一组SQL语句。

事务的基本定义如下：

```
BEGIN TRANSACTION
SQL语句1  /*可以是SELECT,UPDATE, DELETE, INSERT语句*/
[SQL语句2......]
COMMIT/ROLLBACK;
```

事务由BEGIN TRANSACTION和COMMIT/ROLLBACK之间的所有SQL语句组成。其中，BEGIN TRANSACTION表示事务的开始，COMMIT和ROLLBACK表示事务的结束。COMMIT表示事务提交，即事务中所有UPDATE、DELETE、INSERT操作产生的结果已写回磁盘，事务正常结束；ROLLBACK表示事务回滚，即在执行过程中事务因某种故障不能继续执行，那么撤销已经完成的操作，使数据库回滚到事务开始之前的状态。

下面给出转账交易的事务定义。

> [例8.1] 账户U1向账户U2转账50元（事务T0）。
>
> BEGIN TRANSACTION
> SELECT Balance A FROM Deposit WHERE ID='U1' FOR UPDATE;
> A= A - 50;
> IF(A < 0)
> THEN ROLLBACK;
> ELSE
> {UPDATE Deposit SET Balance = A WHERE ID='U1';
> UPDATE Deposit SET Balance = Balance + 50 WHERE ID='U2';
> COMMIT;}

数据的正确性

与查询处理不同，事务包含对数据库的更新、删除和新增操作，这些操作会改变存储在磁盘中的数据，使数据库的状态发生改变。如何保证在事务执行过程中数据库能从一个正确的状态转变为另一个正确的状态，这是事务处理的关键。

通常，导致数据库出现数据异常的情况有两类：

- 事务执行过程中出现系统故障会导致数据丢失和数据不一致的问题。发生系统故障时，如CPU故障、系统断电等，内存中的数据，尤其是数据库缓冲区中的内容会丢失。对于已经执行COMMIT/ROLLBACK语

句的事务，即已提交事务（Committed Transaction），它的部分或者全部执行结果可能仍留在缓冲区中未写回磁盘，那么系统故障就会使得已提交事务对数据库的修改部分丢失或全部丢失，从而使得数据库状态不一致。对于执行了部分SQL语句且未执行COMMIT/ROLLBACK语句的事务，即未提交事务（Uncommitted Transaction）,它对数据库的执行结果可能已写回磁盘，那么系统故障会导致不正确的数据库状态。

- 多个事务同时更新相同的数据，不同事务之间的操作交叉执行会导致数据库状态的不一致。假设，两个用户同时使用账户U2的余额进行购物，用户1支出50元（事务T1）。用户2支出60元（事务2），账户的余额B为100元。两个同时执行时可能存在的操作执行顺序为（这里只关注事务的读写数据操作）：
    - （1）事务T1读取账户余额B，B=100，记作$R_1(B)=100$；
    - （2）事务T2读取账户余额B，B=100，记作$R_2(B)=100$；
    - （3）事务T1修改余额B=B-50，将B=50写回磁盘，记作$W_1(B)=50$；
    - （4）事务T2修改余额B=B-60，将B=40写回磁盘，记作$W_2(B)=40$；

上述执行顺序使得两个事务都执行成功并且最终的账户余额为40元。但是，如果两个事务都执行成功，账户余额应为-10元。在实际中，账户余额不可能为负数，所以两个事务不可能同时执行成功。事务并发执行过程中，不同事务的操作序列调度是随机的。如果按照上面的执行顺序，T1事务的修改丢失了，从而导致了不正确的数据库状态。事务并发执行带来的数据异常现象包括丢失更新（Lost Update）、脏读（Dirty Read）、不可重复读（Non-repeatable Read）、幻读（Phantom Read）、写偏序（Write Skew）和读偏序（Read Skew）。对这部分感兴趣的读者可以阅读其他资料。

事务的ACID性质

事务处理是保证当系统故障和事务并发时数据库状态的正确性。为了实现数据的正确性，数据库管理系统需要保证事务的四个特性：原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）和持久性（Durability）。这四个特性简称事务的ACID特性。

- 原子性：事务中的所有操作要么都执行，要么都不执行，不会出现中间状态。
- 一致性：事务执行的结果必须保证数据库状态的一致性。比如，A账户向B账户转账100元。该事务执行前和事务执行后两个账户的总额度是不发生变化的，并且账户的余额不能为负数。事务的一致性与原子性是密切相关的，甚至有时需要和应用程序配合才能保证。
- 隔离性：多个并发执行的事务相互之间不受干扰，所有的事务像是按某个顺序执行的，彼此之间是隔离的。事务的隔离级别包括读未提交（Read Uncommitted）、读已提交（Read Committed）、可重复读（Repetable Read）、可串行化（Serializable）、快照读（Snapshot）等。
- 持久性：一个事务一旦提交之后，它对数据库产生的数据修改是永久性的并且不能撤销。之后的其他操作或者故障都不会对执行结果产生任何影响。

保证事务的ACID特性主要依靠事务处理技术中的日志恢复机制和并发控制机制，其中日志恢复机制保证事务的原子性和持久性，并发控制机制保证事务的隔离性。