

## **INTRODUCTION:**

A transaction is a logical unit of work that needs to be finished completely or not carried out at all. The University Waste Management System relies on transactional consistency to ensure reliable operations.

### **SCENARIO 1:**

When a worker collects waste from dustbin we will have to maintain a record, where we will insert it into WasteCollection. Also we will reset the fill level of dustbin. In the end, we will update the waste storage quantity. So these are all the properties of transaction.

#### **ACID:**

**Atomicity:** In atomicity we ensure that all operations must execute and succeed together. If updating the storage fail, the collection record must be rolled back.

**Consistency:** We maintain consistency by ensuring that the fill level never exceeds the capacity and the storage quantity cannot be negative.

**Isolation:** The row level locking avoids data conflict in the event that two employees try to collect the same dustbin at the same time.

**Durability:** When a transaction is committed once, then it remains permanently stored even if a system fails.

### **SCENARIO 2:**

Another example of transaction in this system is the monthly payroll processing. The salary will be calculated. The record of each worker will be inserted in the salary table. Also the status of the worker's payroll will be updated.

#### **ACID:**

**Atomicity:** In atomicity we ensure that all operations must execute and succeed together. If updating the status fails and the salary record insertion succeeds then the system will roll back all the transactions.

**Consistency:** We maintain consistency by ensuring that the salary is greater than zero.

**Isolation:** It will ensure that the salary processing operations do not interfere with each other. For instance, if two administrators try to process salary for the same worker at the same time then the row level locking will prevent duplicate salary entries. One transaction must complete before the other begins.

**Durability:** When a transaction is committed once, then it remains permanently stored even if a system fails. For instance, salary data is permanently stored.

### **CONCURRENCY:**

We maintain concurrency by using row level locking and foreign keys to prevent the lost updates and dirty reads. We give the concurrent access to dustbin and salary tables which are controlled by using transactional statements.