Two Phase Transaction Explained

1. Transaction Outbox Pattern:

<https://learn.microsoft.com/en-us/azure/architecture/databases/guide/transactional-outbox-cosmos>

1. SQL Server Transaction Log Architecture

<https://learn.microsoft.com/en-us/sql/relational-databases/sql-server-transaction-log-architecture-and-management-guide?view=sql-server-ver16>

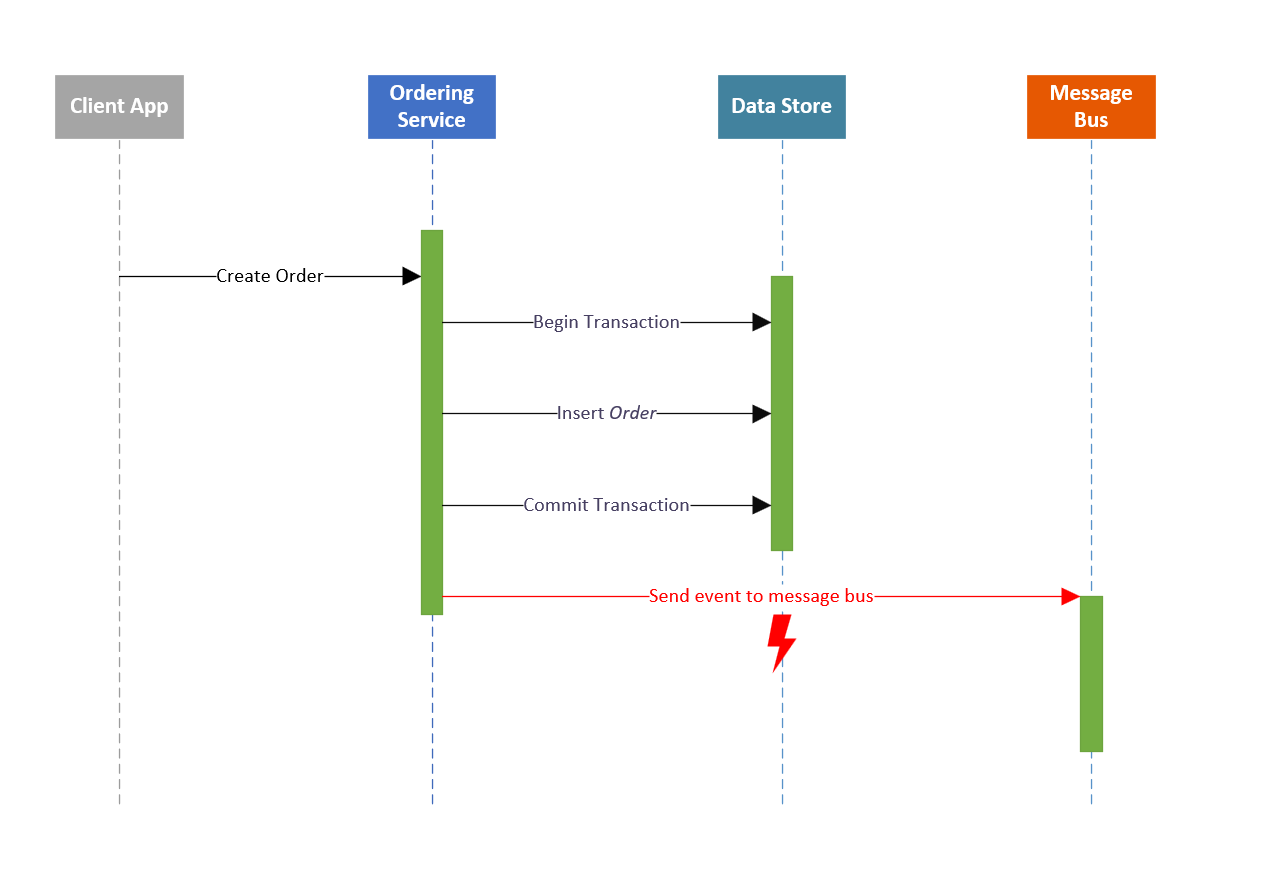
1. They all have the same pattern is there is a Transaction Log before making the final changes:

* In Transaction Outbox pattern for Service Bus Queue, this is the database recording what is going to send to the queue so that later we can use a service to recover
* In SQL Transaction, there is also a service keep writing or synchronizing SQL Transaction Log with the physical database

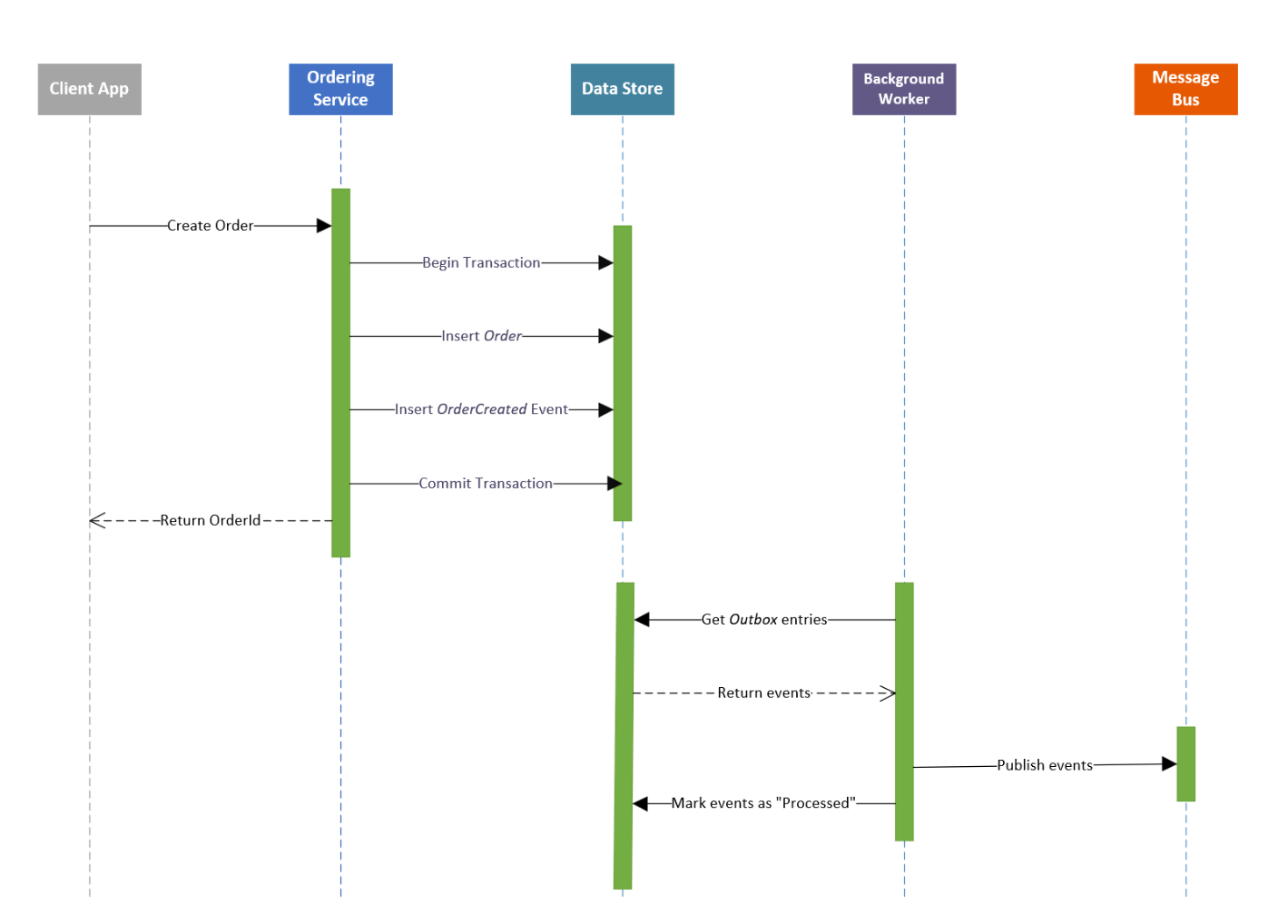
1. What is the relationship between Transaction Log and Commit()

* In the Prepare() step of on a Resource Manager in a 2 phase Transaction Processing, the data regarding to the transaction will be written to Transaction Log, then the resource manager will signal the Commit() call
* For each resource manager, since the transactional data has been save to Transaction Log in a persistent state, it’s stable and a separate service in the resource manager can always find a way to make the Transaction Log and physical queue or database consistent.

1. Sequence diagram of Service Bus Transaction Outbox Pattern:



Step 1: always saves to a Data Store (Transaction Log) first



Step 2: a separate Service (or called Transaction Committer) will always check at the background any uncommitted transaction and commit them. When this proceeds, it always wait for the physical queue or database in synch before delete the Transaction Log.