REPLICATION PART 3

SYSTEM DESIGN

ALTIBASE

ALTIBASE REPLICATION ARCHITECTURE

Replication types

- Lazy – Fast replication that does not affect the master transaction
- Eager – Similar to 2PC(2 Phase commit)

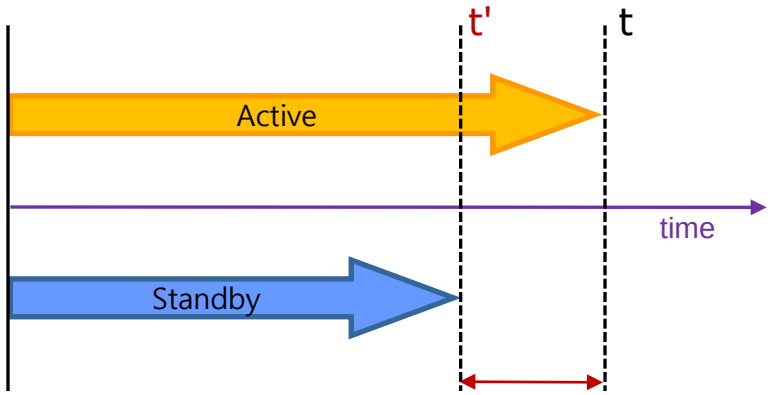
Formation types

- Active-Active
  - Activates ‘Sender’ from every nodes
- Active-Standby
  - System that uses fail-over: Activates all the ‘Senders’ such as Active-Active
  - System for backup only: Sender is activated only from Active server

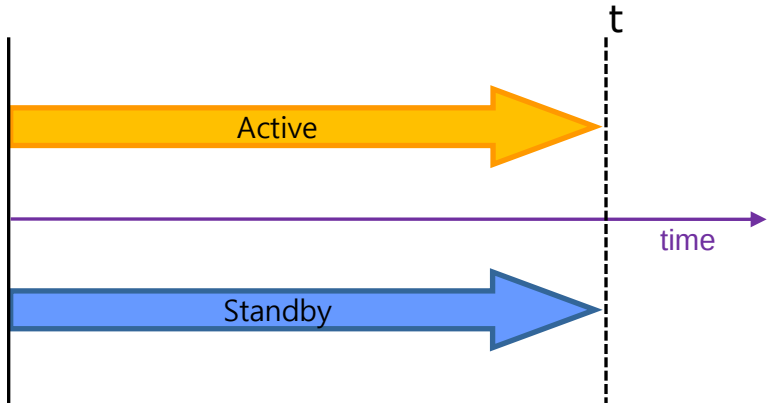
LAZY VS. EAGER

Two different methods depending on the synchronization point

- Lazy(Asynchronous) – The Master transaction and the replication transaction operates separately. The replication is delayed but performance of processing transaction is fast
- Eager(Synchronous) – The Master transaction and the Replication transaction operates as one. The performance is slow but there is no replication delay



[Fig1. lazy]

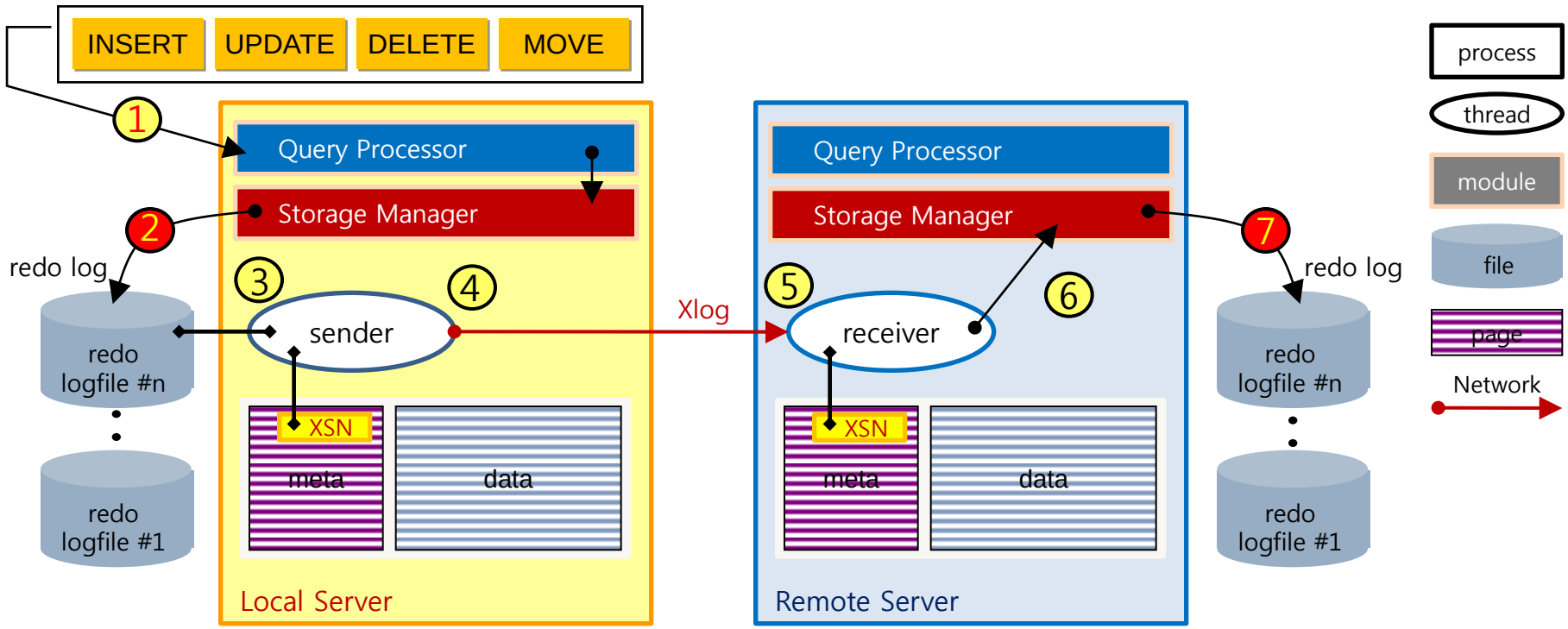


[Fig2. eager]

Main Feature

- There is a trade-off between replication delay or gap and performance

ALTIBASE REPLICATION ARCHITECTURE

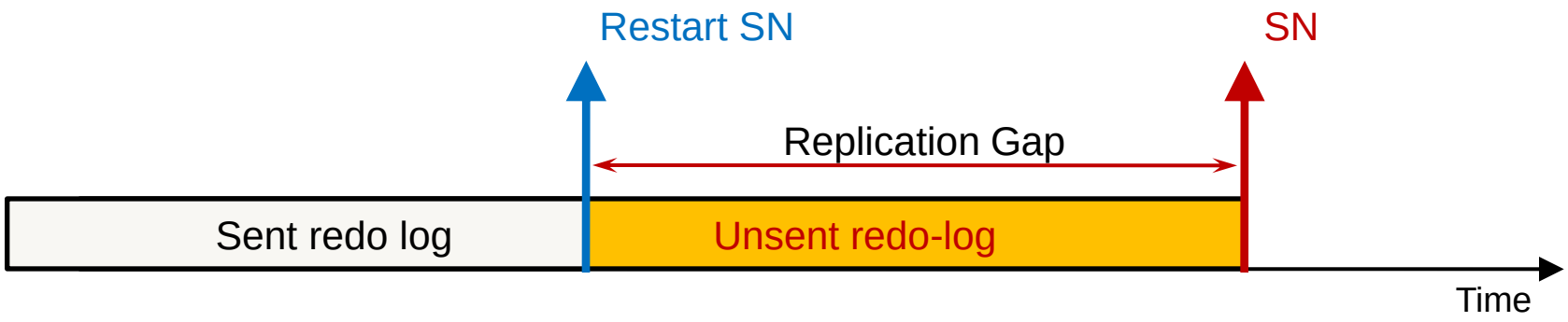


- **Lazy**
  - Master Transaction= 1 + 2
  - Replication Transaction= 3 + 4 + 5 + 6 + 7
- **Eager**
  - Transaction = 1 + 2 + 3 + 4 + 5 + 6 + 7
    - The Master Transaction(2) is confirmed when it's fully applied to Replication transaction(7)

ALTIBASE REPLICATION ARCHITECTURE

Replication Gap

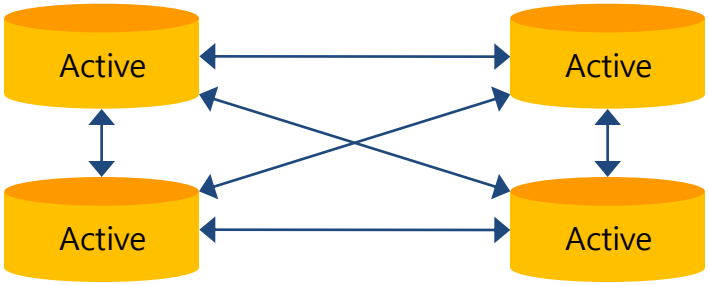
- The replication gap can be checked at a performance view table called v$repgap shown by number
  - Calculated with SN(Sequence Number) the redo log number and Restart SN
  - Replication Gap = [Recent Local Server SN] - [Recent Local Server Restart SN]



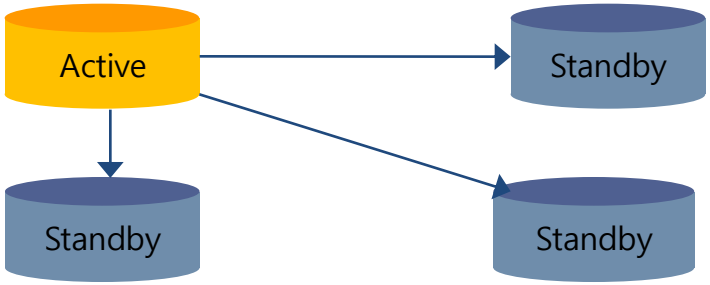
ACTIVE-ACTIVE VS. ACTIVE-STANDBY

Replication Gap

- Different formation depending on the number of DML nodes
  - Active-Active Modification is available from every nodes but there is a possibility of conflicts
  - Active-Standby Modification is available from a particular node but there are no conflicts



[Fig 1. Active-Active]



[Fig 2. Active-Standby]

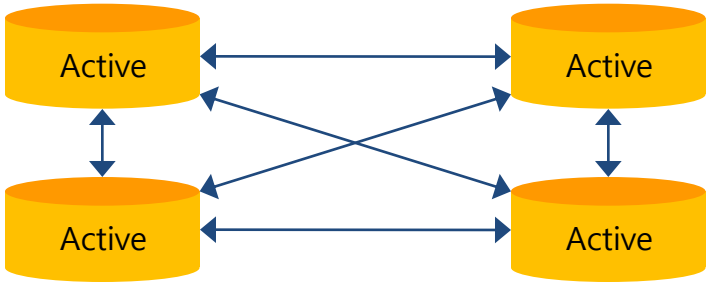
Main Features

- Trade-off between modification conflicts and Load-Balancing
- Consideration for application depending on the formation
  - Active-Active : Lock might be occurred
  - Active-Standby : Maintaining application depending on the nodes role

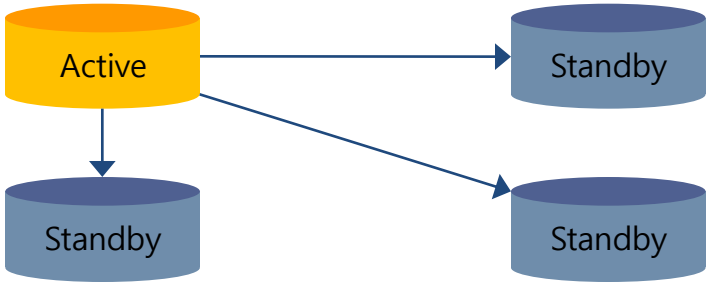
REPLICATION SYSTEM SETTING

How to configure replication system

- Active-Active, Active-Standby(fail-over)
  - Create [Number of entire servers - 1] number of replication objects for each server
  - Sender is activated from all of servers
- Active-Standby(backup)
  - In active server, create replication object as [Number of entire servers - 1]
  - For standby server, create one replication object that corresponds to one active server
  - Sender is activated only from active server



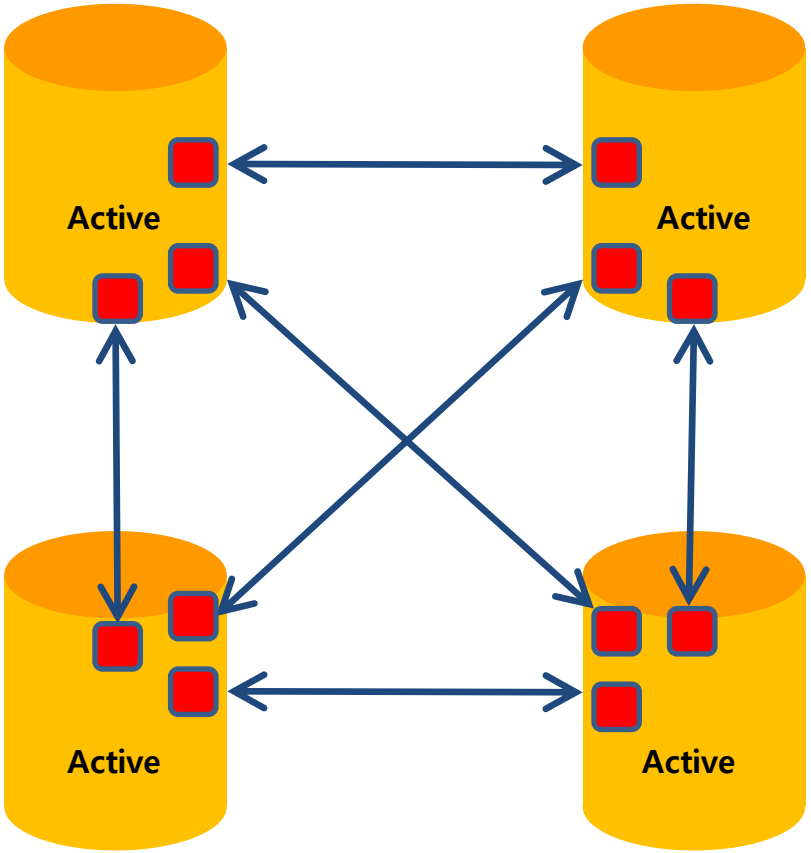
[Fig 1. Active-Active, Active-Standby(fail-over)]



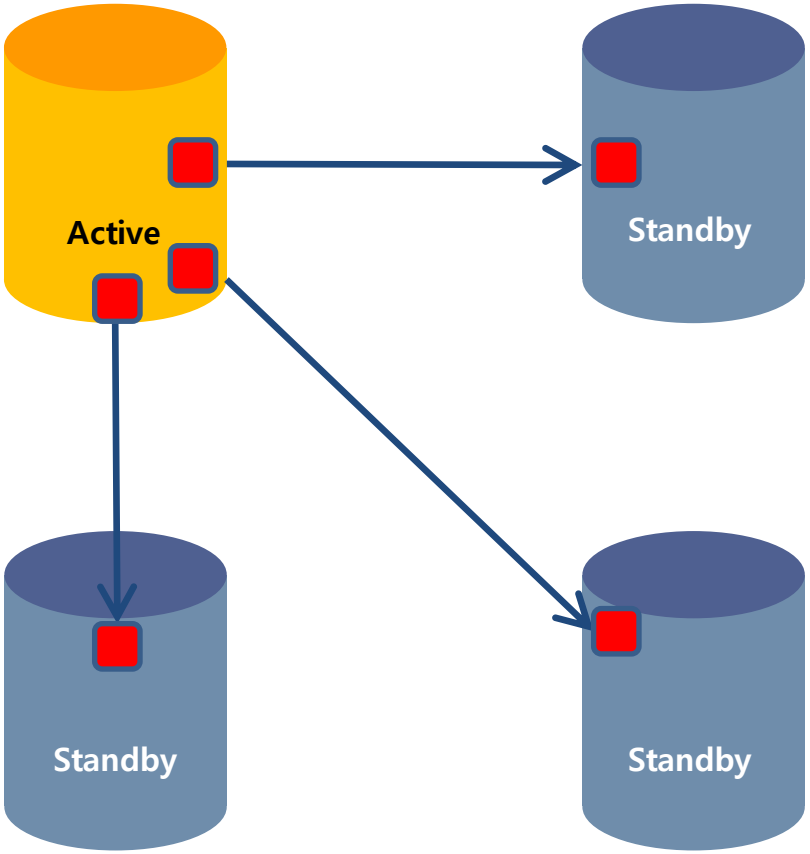
[Fig 2. Active-Standby(backup)]

REPLICATION SYSTEM SETTING

Example of Creating Replication Object



[Fig 1. Active-Active]



[Fig 2. Active-Standby]

Replication Conflict

Replication Conflict

- Conflicts of different (I/U/D) operations between replication servers
- Data inconsistency will be worse

Cause

- Both Lazy type and Active-Active formation are applied together
  - Conflict could occur when it is fail-over even though it's Active-Standby

Solution

- Choose Eager type when data consistency is important than performance
- The best way is to design a system that avoids the such conflicts
  - Design to let each different nodes handle different records
- Conflict Resolution
  - It cannot be fully solved with the provided conflict solutions only

REPLICATION SYSTEM MATRIX

Consideration depends on the formation types and replication types

Categories	Active-Active		Active-Standby	
	lazy	eager	lazy	eager
Replication Performance	Fast	Slow	Fast	Slow
Replication Delay*	Yes	No	Yes	No
Load Balancing	Every DML is available (INSERT,UPATE,DELETE)		Unavailable (Only SELECT)	
Application Program	Lock competes		Lock do not competes	
Replication Conflicts*	Yes	No		
Data Inconsistency	Possible	Never	Possible	Never

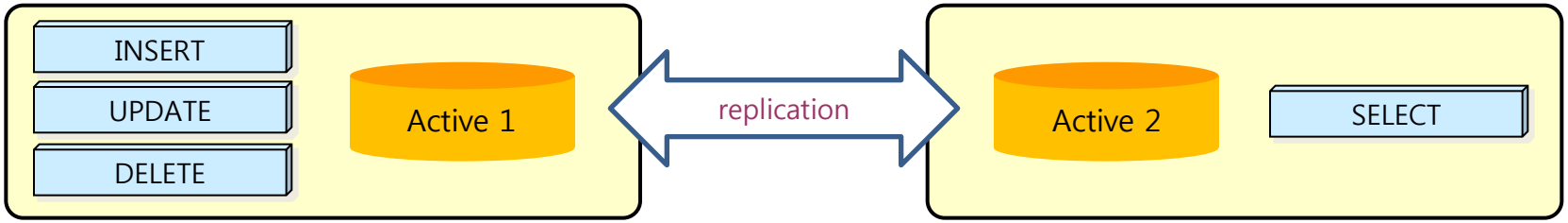
Procedure of Adopting Replication

- Choosing formation types and Replication types that satisfies system
  - Generally Lazy replication type is chosen for fast performance
  - Most ideal architecture of system is a combination of Active-Active & Lazy with no Replication conflicts
- Establishing plan for possible errors depending on the types of formation and Replication
  - **When there is a Network problem*, the recovery plan has to be established even though the replication type is eager**

REPLICATION SYSTEM DESIGN

Assigning server that does DML only

- Design server1 for DML only and server2 for SELECT only



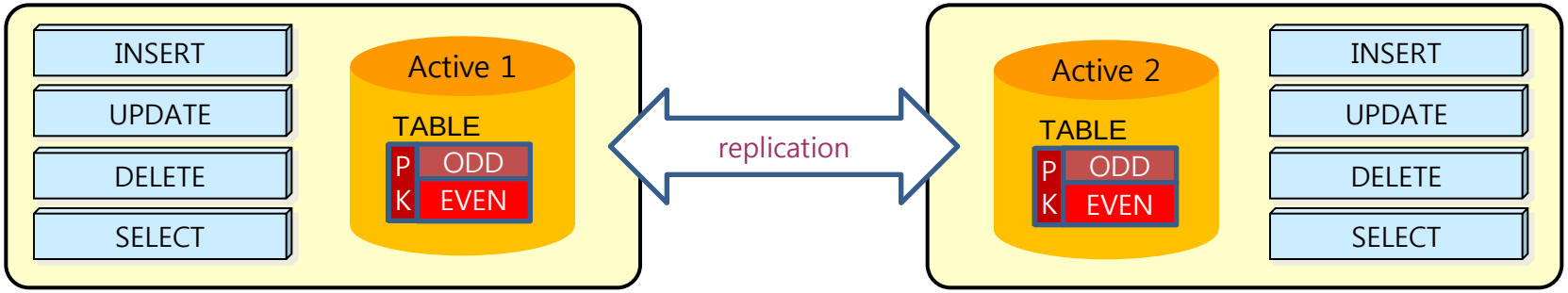
Feature

- Load-balancing is only allowed in SELECT

REPLICATION SYSTEM DESIGN

Divide a number of PKs same as number of nodes

- Design server1 for odd number only and server2 for even number only



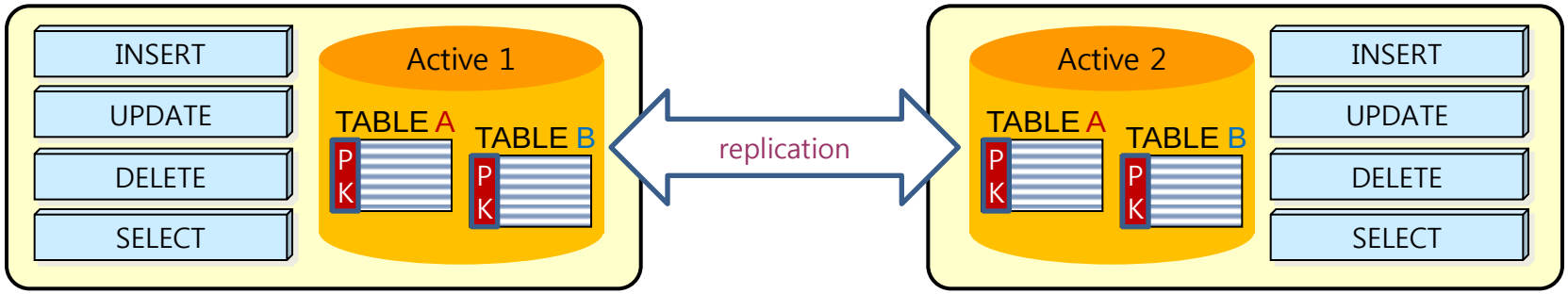
Main feature

- Load-balancing of all the DML operations is possible but caution is advised when setting up the application program

REPLICATION SYSTEM DESIGN

Dividing table according to its tasks

- Server 1 is responsible for table A's DML only and server 2 is responsible for table B's DML only



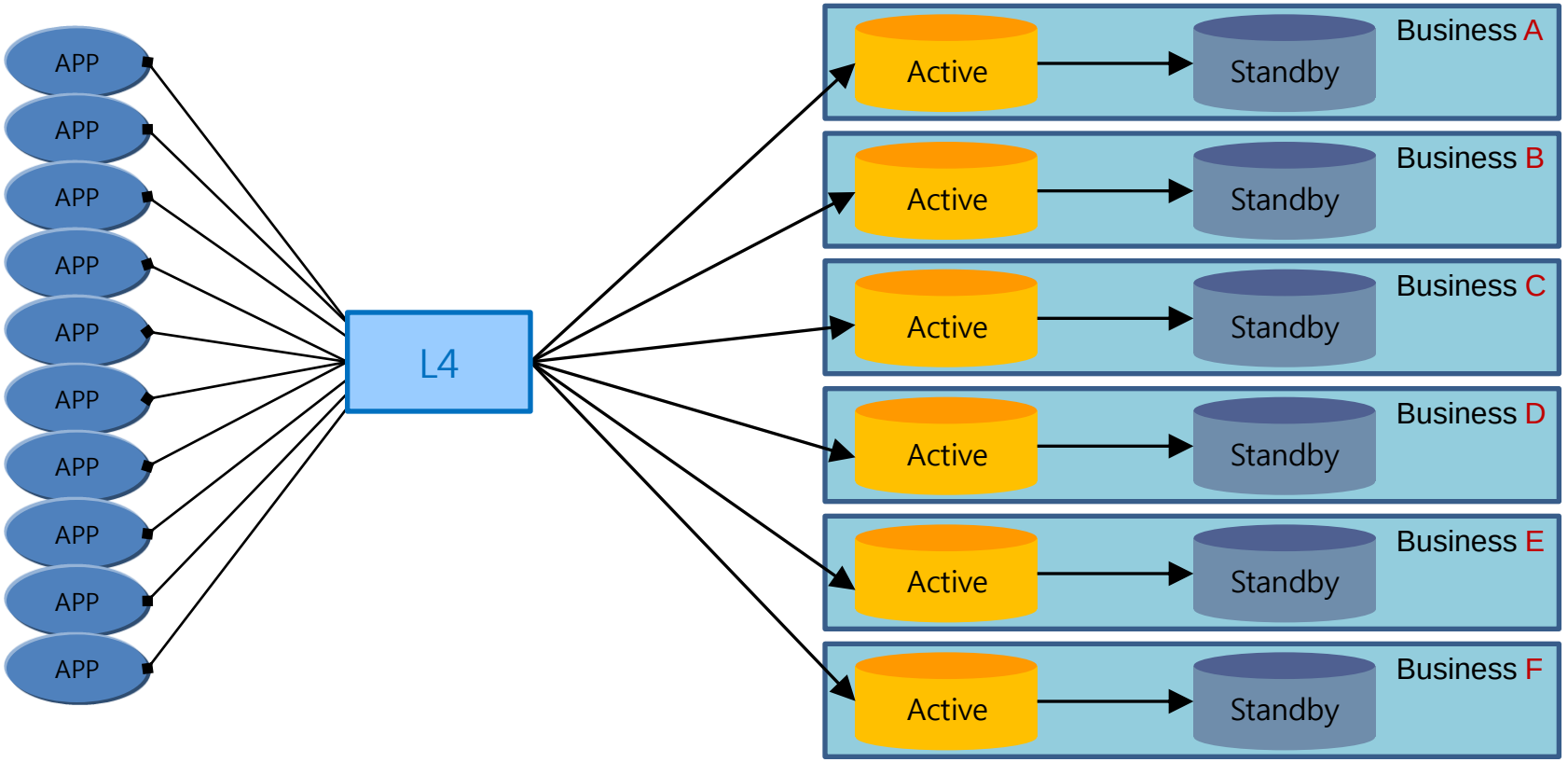
Main feature

- Extra consideration is needed when processing complex tasks even though the load-balancing of DML operation is possible

REPLICATION SYSTEM DESIGN EXAMPLE

❖ **Example of system design that assigns the server for DML only using L4**

- ♦ Connecting to server by identifying application program with IP from L4



Q & A

ALTIBASE

Thank you!

Altibase Education Center

Tel : 02-2082-1451
Fax : 02-2082-1459
E-mail : education@altibase.com
Homepage : http://edu.altibase.com