

Performance Characterization of Oracle JD Edwards EnterpriseOne with Microsoft SQL Server Clustering

Performance Characterization of Microsoft SQL Server Failover Cluster instance and Always on availability groups during planned failover scenarios.

October, 2025, Version [\[1.0\]](#)
Copyright © 2025, Oracle and/or its affiliates
Public

Purpose Statement

Microsoft SQL Server Clustering provides high availability and business continuity. This document provides an overview of testing Oracle JD Edwards EnterpriseOne with Microsoft SQL Server Clustering to provide a better understanding of clustering of Microsoft SQL server and the behavior of JD Edwards EnterpriseOne when clustering is used. This document describes the features and enhancements included in Release 26 (Tools Release 9.2.26.0). It is intended solely to help you assess the business benefits of upgrading to Release 26 (Tools Release 9.2.26.0) and planning for the implementation of Microsoft SQL Server Clustering.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Purpose statement	2
Disclaimer	2
Executive Summary	4
Overview	5
Configuration Overview	5
Microsoft SQL Server High Availability Concepts	5
Introduction	5
Failover Cluster Instance (FCI)	5
Always on Availability Group (AG)	6
Workload Profiles	6
Testing Methodology	6
Performance Characterization	7
Interactive Applications Results	7
Batch Processes Results	8
Failover Cluster Instance vs Always on Availability Group (Read-Write Only)	9
Failover Recovery	10
Conclusion	10
Appendix A	12
Metrics	12
Use Cases	12
Interactive Application Use Case	13
Batch Processes Use Case	13
Appendix B	14
Batch Job Failures in Always on Availability Group Read-Only Mode – Root Cause and Performance Analysis	14
Appendix C	14
Configuring ODBC and Microsoft SQL Application Intent in JD Edwards EnterpriseOne	14
Objective and Parameter	14
Step-by-Step Instructions:	15
Appendix D	16
System Architecture	16
Failover Cluster Instance (FCI)	16
Availability Group (AG)	17

Executive Summary

In today's competitive world, customers expect businesses to be available to offer their services whenever they need. This translates into businesses having to run their operations round the clock on every day of the year to make it possible. The backbone of such operations are systems with robust technologies that are highly available to provide the required business continuity. These technologies include Web Servers, Application Servers, and Databases. The high availability and disaster recovery capabilities of these technologies enable businesses to run their operations 24x7.

One of the key database technologies that JD Edwards EnterpriseOne customers use is Microsoft SQL Server. Microsoft SQL Server provides clustering capabilities for high availability and disaster recovery such as Failover Cluster Instance (FCI) and Always On availability groups. Microsoft SQL Server takes advantage of the Windows Server Failover Cluster (WSFC) services and capabilities to support Always On availability groups and SQL Server failover cluster instances. If a cluster node or service fails, the failover scenario is initiated and services that were hosted on that node is automatically or manually transferred to another available node. For more details, see [Windows Server Failover Clustering](#).

Microsoft SQL Server clustering ensures that the database is always available for transactions from JD Edwards EnterpriseOne, eliminating business downtime. With Release 26 (Tools Release 9.2.26.0), JD Edwards EnterpriseOne customers, using Microsoft SQL Server as the database, can take advantage of Microsoft SQL Server clustering capabilities and benefit from increased application availability, better returns on hardware investments, and simplified high availability deployment and management.

This document provides an overview of the performance characterization of JD Edwards EnterpriseOne with Microsoft SQL Server Failover cluster instance and Always On availability group highlighting the capabilities and benefits of using Microsoft SQL Server clustering. It also outlines testing approach for Microsoft SQL Server, key performance metrics analyzed, common issues identified, and best practices recommended for ensuring optimal database performance and stability in enterprise environments.

Overview

The purpose of this document is to measure and compare the performance of Microsoft SQL Server deployed in two high-availability architectures, ensuring alignment with JD Edwards EnterpriseOne performance objectives.

- Failover Cluster Instance (FCI)
- Always On Availability Group

The primary focus is to validate interactive and batch workload performance during steady-state operation and following failover events, ensuring compliance with JD Edwards EnterpriseOne performance acceptance criteria. The evaluation provides insights into workload efficiency, failover recovery behavior, and architectural suitability for enterprise-scale deployments.

Configuration Overview

Failover Cluster Instance (FCI)

The FCI architecture provides node-level high availability using shared storage. Only one node (replica) is active at a time. All database operations in FCI run in a single operational mode (Read-Write), ensuring transactional consistency but limiting scalability to one active node.

Availability Group (AG)

The AG architecture offers database-level replication, enabling multiple replicas across nodes for high availability and disaster recovery. It supports two operational modes:

- AG (Read-Write) – the primary replica servicing both reads and writes.
- AG (Read-Only) – secondary replica serving only read operations.

Microsoft SQL Server High Availability Concepts

Introduction

To better understand the results, it's important to define the core Microsoft SQL Server high availability options evaluated in this report for Failover Cluster Instance (FCI) and Always On availability group and clarify their key differences and operating modes.

Failover Cluster Instance (FCI)

An FCI is a high-availability configuration that provides failover support for an entire SQL Server instance. FCIs use Windows Server Failover Clustering (WSFC) and shared storage. In the event of a node failure or planned maintenance, the entire instance (including system and user databases) fails over to another node in the cluster. Because of the shared storage, only one node owns the instance and its databases at a time.

Key Characteristics:

- Failover occurs at the instance level.
- Requires shared storage (for example, SAN, Storage Spaces Direct).
- Only one node is active at a time.
- Offers robust availability but may incur longer failover durations.

Always On Availability Group

An AG allows for high-availability and disaster recovery at the database level. Introduced in SQL Server 2012, AGs allow for database-level failover and provide high availability for a set of user databases. Unlike FCI, AGs do not require shared storage and support multiple readable secondary replicas for offloading read-only workloads.

Key Characteristics:

- Database-level failover.
- Does not require shared storage.
- Supports multiple replicas (primary and secondary).
- Can offload read-only traffic to secondary replicas (if configured).

Read-Write vs. Read-Only Modes

In the context of Availability Groups and FCIs:

- **Read-Write Mode:** Refers to the primary replica or instance where both read and write operations are allowed. This is the active node responsible for processing OLTP (Online Transaction Processing) workloads.
- **Read-Only Mode:** Typically refers to secondary replicas in an Availability Group that are configured to accept read-only queries. These replicas can be used to offload reporting workloads, thus reducing the load on the primary.

More information on FCI and AG configurations can be found in the official [Microsoft SQL Server Documentation](#).

Workload Profiles

The workloads used in this testing represent real-world enterprise scenarios, specifically tailored to simulate JD Edwards EnterpriseOne application usage patterns. These include both interactive applications and batch processing operations.

Table1. [SQL Server Workload Profiles](#)

WORKLOAD TYPE	DESCRIPTION	EXAMPLE TRANSACTIONS
Interactive Applications	Real-time UIs, query interfaces	Supplier Ledger Inquiry, Sales Order Entry
Batch Processes	Scheduled, background jobs	Sales Order Invoicing, Work Order Processing

Table 1 categorizes the primary types of workloads used during Microsoft SQL Server performance testing. Each workload type reflects typical usage patterns and transaction types within enterprise systems, allowing for more targeted and realistic performance evaluation.

Testing Methodology

The testing methodology, summarized in Table 2, is to show the actual operational conditions, including planned failover events. Metrics were gathered across various Microsoft SQL Server high-availability configurations and compared against predefined performance benchmarks to assess system behavior under load and failover scenarios.

Table2. SQL Server Testing Methodology Use Case

USE CASE	DESCRIPTION	TARGET
Interactive Application	Simulates user transactions through JD Edwards EnterpriseOne	< 100 ms response time
Batch Processes	Runs scheduled data operations	± 5% runtime deviation
Failover Scenario	Induced node failure and monitoring recovery time	≤ 5 min service restoration

Performance Characterization

This section presents the consolidated performance test results of Microsoft SQL Server under Failover Cluster Instance and Always On availability group configurations. The analysis focuses on two primary workload types: interactive application latency and batch job runtimes during planned failover scenarios.

The reported results reflect the performance data between before-failover and after-failover states, highlighting how each high-availability configuration behaves during planned failover events. Rather than showing raw performance figures, the emphasis is on relative performance impact and recovery behavior, enabling a clear comparison between FCI and AG architectures under real-world conditions.

Interactive Applications Results

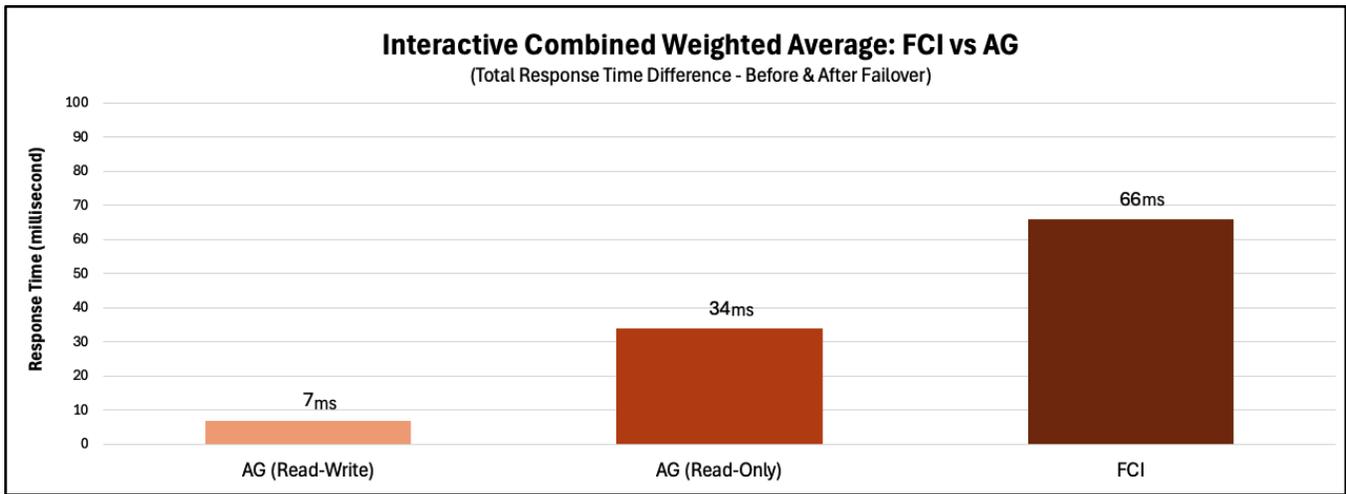
The metrics collected for interactive applications represent the average response time difference (in milliseconds) measured before and after failover across different Microsoft SQL Server high-availability configurations, namely Failover Cluster Instance (FCI) and Always On availability group (AG) in both Read-Write and Read-Only modes.

The purpose of this analysis is to determine whether each configuration maintains user-facing transaction response times within the JD Edwards acceptance criterion of 100 milliseconds after a planned failover event.

The EnterpriseOne interactive workload suite was used to simulate user transactions such as form loads, data retrieval, and data commit operations. This ensures that the response-time measurements reflect realistic application and middleware performance under production-like conditions.

Figure 1 represents the total difference in average response time (in milliseconds) measured between the before-failover and planned post-failover states for each configuration. It compares the response-time behavior across the three configurations tested and highlights how architectural differences between Failover Cluster Instance (FCI) and Always On availability group influence the responsiveness of interactive workloads.

Figure1. SQL Server Interactive Application Results



AG (Read-Write) achieved the lowest response-time difference of 7ms, demonstrating highly efficient recovery with minimal latency variation during failover. AG (Read-Only) recorded 34ms, remaining well within the JD Edwards 100ms acceptance threshold. FCI measured 66ms, showing consistent and predictable behavior across failover transitions.

Overall, all configurations met the JD Edwards EnterpriseOne performance criterion, confirming that both AG and FCI maintain acceptable response times during failover operations.

Batch Processes Results

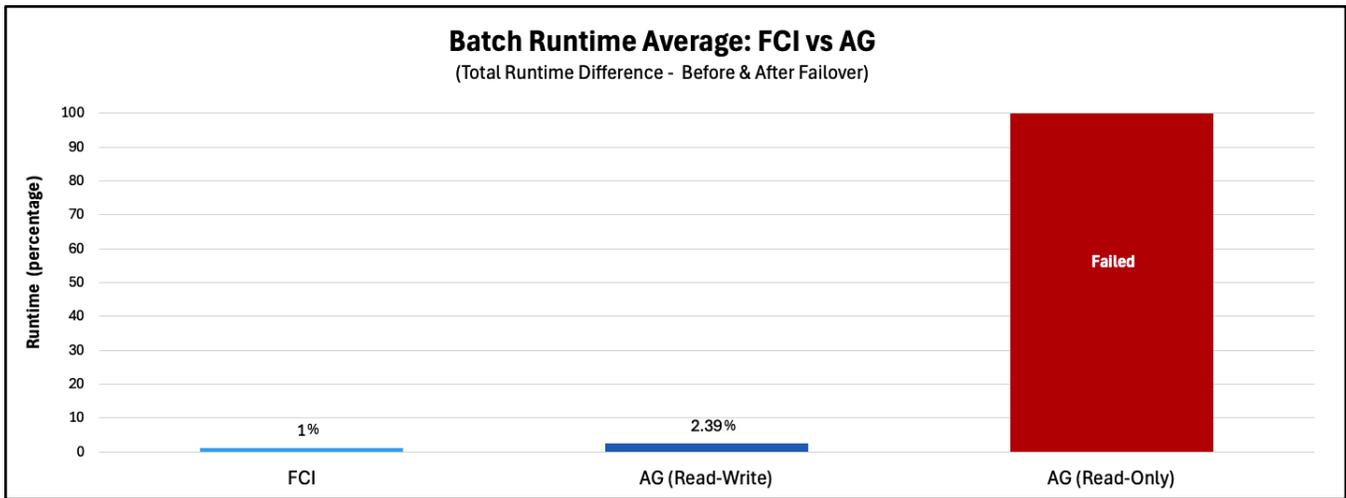
The metrics presented for batch processes reflect the total runtime difference measured before and after failover for EnterpriseOne batch job executions across Microsoft SQL Server high-availability configurations, namely Failover Cluster Instance (FCI) and Always On availability group (AG), tested in both Read-Write and Read-Only modes.

The objective of tests is to verify that each configuration maintains batch job runtime consistency within the ±5% JD Edwards performance acceptance criteria, following a planned failover event.

Standard EnterpriseOne batch workloads were used, simulating typical production activities such as data extraction, transactional updates, and commit operations. This realistic workload design ensures that the measured differences accurately reflect the system’s performance and stability under high-availability conditions.

Figure 2 illustrates the percentage difference in total batch job runtimes, comparing before-failover and after-failover performance across each configuration. The results highlight each architecture’s ability to deliver runtime stability through failover transitions.

Figure2. SQL Server Batch Processes Results



FCI demonstrated the most consistent performance, with a runtime difference of less than 1%, well within the $\pm 5\%$ JD Edwards EnterpriseOne performance acceptance criteria. AG (Read-Write) also met the performance target, showing a 2.39% difference, indicating steady runtime behavior through failover.

However, the AG (Read-Only) configuration is unsuitable for transactional batch workloads requiring timely data consistency, due to latency in synchronization between the primary (where updates happen) and the secondary (where selects are issued). For fast-executing jobs, the secondary may lag behind, resulting in stale or inconsistent reads, such as missing job statuses or intermediate data. While suitable for read-only reporting, this configuration introduces consistency risks in time-sensitive or state-dependent processes.

Overall, both FCI and AG (Read-Write) configurations maintained the performance within the $\pm 5\%$ acceptance limit, confirming their suitability for JD Edwards EnterpriseOne batch processing in high-availability SQL Server environments.

Failover Cluster Instance vs. Always on Availability Group (Read-Write Only)

The comparative analysis between Microsoft SQL Server Failover Cluster Instance (FCI) and Always On availability group (AG) in Read-Write mode was conducted to evaluate relative performance changes before and after a planned failover, under identical JD Edwards EnterpriseOne workloads.

Both configurations were tested using interactive and batch operations, focusing on:

- Interactive response time changes
- Batch job runtime deviation
- Failover recovery duration

This analysis highlights how architectural differences between FCI and AG influence system responsiveness and workload stability after failover. While FCI offers node-level redundancy through shared storage and WSFC, AG supports database-level high availability, with benefits in read scalability and transactional latency, particularly in the Read-Write (primary) replica.

Table 3 summarizes the absolute difference between before-failover and after-failover measurements for each configuration.

Table 3. SQL Server FCI and AG Comparative Analysis

METRIC	FCI	AG (READ-WRITE)	DIFFERENCE
Interactive Response	66 ms	7 ms	AG faster by 59 ms
Batch Runtime Δ	< 1%	2.39%	Slightly higher deviation
Failover Recovery	5 min	5 min	Same

Interactive Workloads:

As shown in Table 3, the response time difference between before and after failover is much smaller for AG (Read-Write) (7 ms) compared to FCI (66 ms). Both values remain well below the JD Edwards acceptance criterion of 100 ms, indicating that AG provides more consistent and lower latency across failover events, because of architectural advantages like local commit handling and reduced coordination overhead.

Batch Workloads:

Table 3 also highlights that AG (Read-Write) experiences a 2.39% change in batch job runtime after failover, which is slightly higher compared to the less than 1% change observed for FCI. Despite this, both configurations remain well within the JD Edwards ±5% acceptance threshold for batch runtime variation.

Failover Recovery

Both FCI and AG (Read-Write) configurations successfully restored full service within five minutes of the planned failover, demonstrating robust operational readiness for high-availability environments.

As detailed in Table 3, the magnitude of performance changes caused by failover differs between the two architectures:

- AG (Read-Write) delivers significantly more stable interactive response times post-failover.
- FCI exhibits slightly more consistent batch job runtimes.

Overall, both solutions provide reliable failover recovery with performance impacts well within acceptable thresholds.

Conclusion

Microsoft SQL Server Failover Cluster Instance (FCI) and Availability Group (AG – Read-Write) configurations successfully meet the JD Edwards EnterpriseOne performance criteria for both interactive and batch workloads. Failover recovery times were consistently under five minutes, validating high availability.

FCI offers stability and simplicity, making it ideal for environments prioritizing reliability.

AG (Read-Write) provides faster response times and supports read scalability, making it suitable for performance-sensitive workloads.

However, AG Read-Only mode is not viable for batch processing, due to its inability to handle the replication latency, secondary replicas may not reflect the most current state of the data, which can lead to inconsistent results or failure during execution of batch logic that depends on up-to-date transactional data. required write operations. This configuration requires further technical remediation and validation before adoption for transactional workloads.

Appendix A

Metrics

A metric is a unit of measurement used for evaluation and comparison.

- Interactive-User Response Time Metric** - Collected using the Apache JMeter tool, these metrics track the timing of HTML-based user actions replayed through JMeter. The tool interfaces with the EnterpriseOne architecture through the web to gather server-side response times. After execution, JMeter generates a summary HTML report containing all the relevant timing data, which is then extracted for analysis.
- Batch Runtime Metric** - Metrics for batch processes focus on two key areas - the total processing duration and the rate of successful completion for Universal Batch Engines (UBEs).

Use Cases

The following interactive applications and batch processes were utilized to test the Microsoft SQL Server environments:

Table4. Analysis Interactive Applications and Batch Processes involved in the Testing.

INTERACTIVE APPLICATIONS	DESCRIPTION
L0411I	Supplier Ledger Inquiry
L3411AE	MRP Messaging WO Orders
L3411BE	MRP Messaging OP Orders
L3411CE	MRP Messaging OT Orders
L4310E	Purchase Order Entry
L17500E	Case Management Add
L31114U	Work Order Completion
L42101E	Sales Order Entry
L42101U	Sales Order Update
GoToEnd	Address Book - GoToEnd
BATCH PROCESSES	DESCRIPTION
R42565	Sales Order Invoicing
R31410	Work Order Processing
R3483	MRP Processing
R43500	Purchase Order Print

Interactive Application Use Case

To evaluate the performance of JD Edwards EnterpriseOne interactive applications, a suite of automated HTML scripts was used. These scripts simulate real user interactions by performing typical tasks through the web-based interface of JD Edwards EnterpriseOne. The simulations are executed using Apache JMeter, a performance testing tool designed to measure system behavior under various loads.

JMeter acts as a virtual user, sending requests through the JD Edwards EnterpriseOne web URL interface and recording response times for each action. These actions mimic real-world usage scenarios such as logging in, navigating menus, opening applications, entering data, and submitting transactions.

The main objectives of these use cases are to:

- Measure server-side response times for different interactive actions.
- Identify potential performance bottlenecks in the application flow.

When the scripts are executed, JMeter generates a Summary Report that includes detailed timing data for each simulated user action. This report is then analyzed to assess overall system responsiveness and stability under test conditions.

Batch Processes Use Case

A batch process, also known as a Universal Batch Engine (UBE), is a non-interactive job submitted to the JD Edwards EnterpriseOne Logic Server. These processes are initiated either through the command line or through the HTML Server web interface, using predefined processing options and data selection criteria. When submitted, they are placed into batch queues for execution.

EnterpriseOne batch processes vary in:

- Execution duration (from quick jobs to long-running processes)
- Processing characteristics (such as, business logic-heavy versus database-intensive)
- Output volume and format (such as the size and generation time of PDF reports)

Testing a broad range of batch jobs helps illustrate how various workloads impact the performance and behavior of different components within the JD Edwards EnterpriseOne architecture.

The tests were conducted using Microsoft SQL Server as the backend database platform, to measure how batch processes interact with and perform against this environment. Several high-volume batch processes were executed as part of this evaluation:

- Work Order Processing (R31410): Processed 38,898 work orders
- Sales Order Invoicing (R42565): Processed 7,252 sales orders
- MRP Processing (R3483): Processed 50,001 sales, purchase, and work orders
- Purchase Order Print (R43500): Processed 255,481 purchase orders in a single run

These tests provided insight into how batch processing workloads scale and perform within an JD Edwards EnterpriseOne deployment using Microsoft SQL Server, including database interaction, processing efficiency, and output generation.

Appendix B

Batch Job Failures in Always On Availability Group Read-Only Mode – Root Cause and Performance Analysis

During the performance characterization of Microsoft SQL Server environments, a consistent failure of batch jobs was observed when running under Availability Group (AG) Read-Only mode. This behavior stems from the way Microsoft SQL Server handles read and write operations in high-availability configurations:

- In Read-Only mode, all DQL (SELECT) queries are routed to the secondary replica.
- DML (INSERT, UPDATE, DELETE) and DDL (CREATE, ALTER, and so on) operations must be executed on the primary replica.
- Batch jobs in EnterpriseOne often include write operations such as:
 - Logging progress
 - Updating job statuses
 - Creating temporary or staging tables

Due to replication latency, secondary replicas may not reflect the most current state of the data, which can lead to inconsistent results or failure during execution of batch logic that depends on up-to-date transactional data.

Appendix C

Configuring ODBC and Microsoft SQL Application Intent in JD Edwards EnterpriseOne

To configure the ODBC (Enterprise Server) and Microsoft SQL (JAS Server) Application Intent parameter for JD Edwards EnterpriseOne in order to control routing of read-only workloads to readable secondary replicas (such as in Always On availability groups for Microsoft SQL Server).

Use Case Scenario

Table5. ODBC and Microsoft SQL Application Intent Setting

SCENARIO	RECOMMENDED SETTING
Reporting workloads on secondary	Read-Only
All transactions on primary	None (default)

Objective and Parameter

- Setting Name: ODBC Application Intent / Microsoft SQL Application Intent
- Value: None (default – all queries go to primary replica)
- Alternative Value: Read-only (if connecting to a readable secondary for reporting or similar use cases)
- Ensure the SQL Server is part of an Always On availability group with readable secondary replicas.
- You must have admin access to the ODBC data source configuration used by JD Edwards EnterpriseOne.

Step-by-Step Instructions:

Step 1: Open JD Edwards Server Manager.

- Log in to the JD Edwards Server Manager Console.
- Navigate to the Managed Home (for example, HTML Server or Enterprise Server) where the ODBC/ Microsoft SQL is configured.

Step 2: Locate the Database Connection Settings.

- Go to the Configuration tab.
- Expand the Database section to view the ODBC / Microsoft SQL connection parameters.

Step 3: Modify ODBC Application Intent.

- Locate the setting: ODBC Application Intent / Microsoft SQL Application Intent.
- From the drop-down, select one of the following:
 - Read-only – To route read-only queries to a readable secondary.
 - None – Default behavior, all traffic goes to the primary replica.

Step 4: Save and Restart.

- Click **Save** to apply the changes.
- Restart the JD Edwards service(s) associated with the modified ODBC / Microsoft SQL source to apply the new settings.

Figure3. Configuring ODBC Application Intent Value

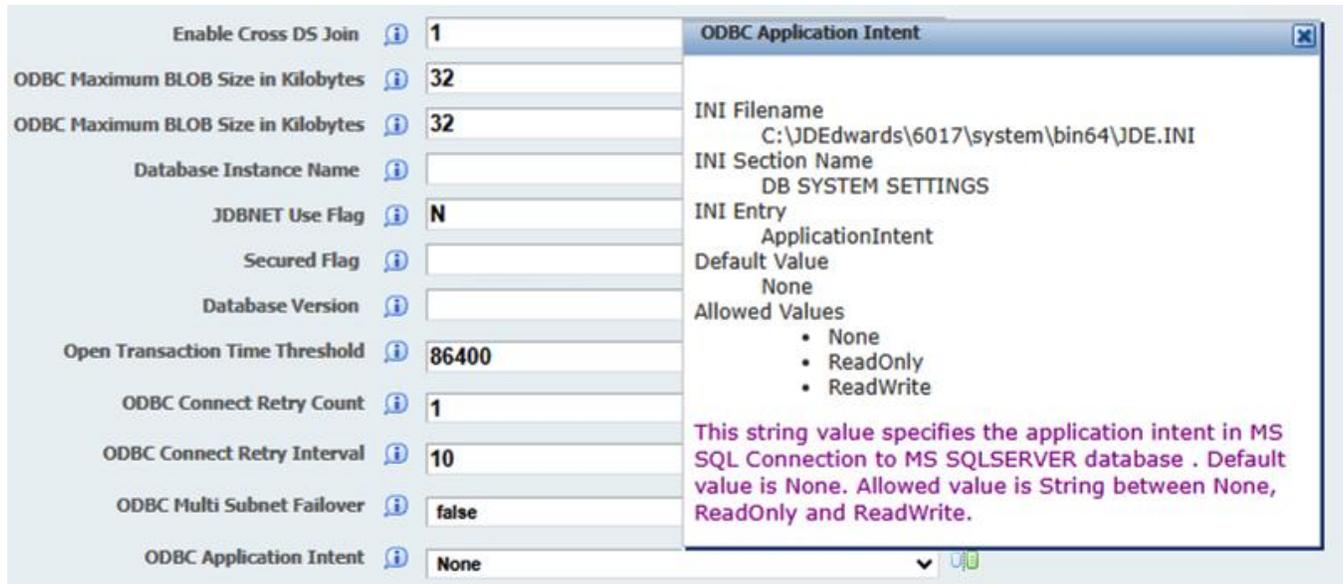
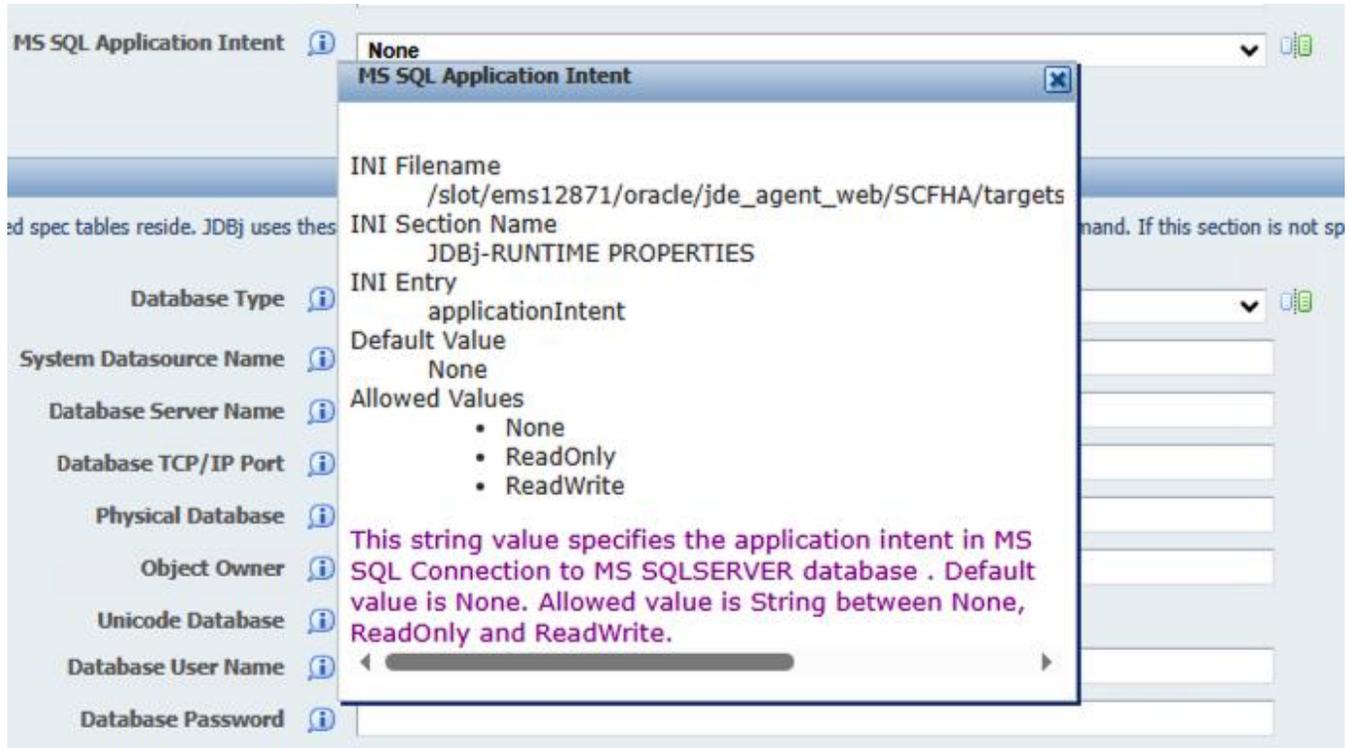


Figure4. Configuring SQL Application Intent Value



Appendix D

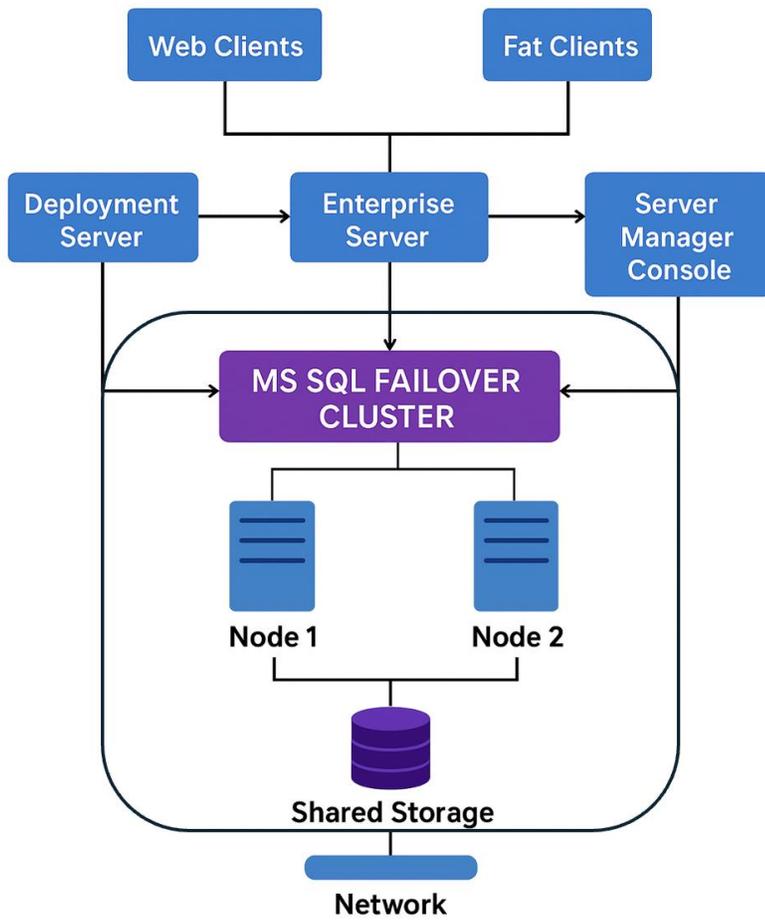
System Architecture

Failover Cluster Instance (FCI)

Single active SQL instance fails over seamlessly to the passive node using shared storage. Supports high availability for both reads and writes.

- Shared storage configuration.
- Automatic failover handled at the cluster service level.
- Single database instance across nodes.

Figure5. JD Edwards EnterpriseOne Architecture with SQL Server Failover Clustering (FCI)

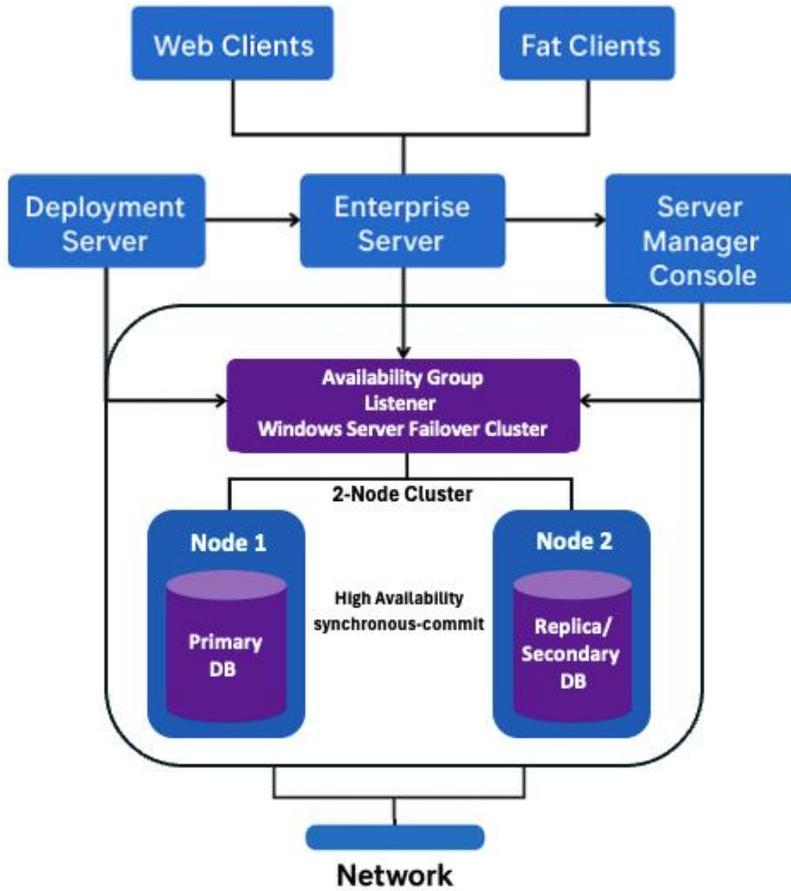


Availability Group (AG)

AG enables multiple replicas: one for read-write (primary) and one or more for read-only workloads (secondary). Supports high availability, disaster recovery, and reporting scaling.

- Database-level replication across primary and secondary nodes.
- Supports both **Read-Write (Primary)** and **Read-Only (Secondary)** configurations.
- Designed for high availability (HA) and disaster recovery (DR).

Figure6. JD Edwards EnterpriseOne Architecture with SQL Server Always on Availability Group (AG)



Connect with us

Call **+1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

 blogs.oracle.com

 facebook.com/oracle

 twitter.com/oracle

Copyright © 2025, Oracle and/or its affiliates. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.