Oracle® Cloud

Using Oracle TimesTen In-Memory Database for Oracle Cloud Infrastructure Kubernetes Engine

Latest Cloud Release
G33801-01
September 2025

ORACLE

Oracle Cloud Using Oracle TimesTen In-Memory Database for Oracle Cloud Infrastructure Kubernetes Engine, Latest Cloud Release

G33801-01

Contents

About This Content

This document provides background information to help you understand how the Oracle TimesTen In-Memory Database for Oracle Cloud Infrastructure Kubernetes Engine (TimesTen Database for OKE) stack listing in Oracle Cloud Marketplace work. It also provides step-by-step instructions and examples that show how to deploy the TimesTen Kubernetes Operator (TimesTen Operator) using the stack and other common tasks.

Audience

This document is intended for users of Oracle Database, TimesTen, and Kubernetes.

To work with this document, you should be familiar with Oracle Cloud, Kubernetes Engine (OKE), TimesTen, SQL (Structured Query Language), and database operations.

Related Resources

See these Oracle resources:

- TimesTen documentation
- Kubernetes Engine documentation
- Marketplace documentation
- Resource Manager documentation

Conventions

The following text conventions are used in this document.

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Get Started

Oracle TimesTen In-Memory Database (TimesTen) is a relational database that resides entirely in physical memory—which is, at the same time, completely persistent and recoverable. You can deploy TimesTen as a standalone database or as cache to a back-end Oracle database. The TimesTen Kubernetes Operator (TimesTen Operator) is a Kubernetes operator that you can use to create, manage, and monitor TimesTen databases in a Kubernetes cluster. The TimesTen Operator configures TimesTen databases, users and schemas, and if applicable, cache and replication. The TimesTen Operator automatically manages and monitors many TimesTen databases simultaneously in your namespace. For example, in an active standby pair replication scheme, if the active database fails, the TimesTen Operator automatically promotes the standby database to active. Then, Kubernetes replaces the Pod that contains the failed active database. Finally, the TimesTen Operator brings up a new standby database in the replacement Pod. The TimesTen Operator performs all these operations automatically without need of user intervention.
Oracle Cloud Infrastructure Kubernetes Engine (Kubernetes Engine or OKE) is a fully managed, scalable, and highly available Kubernetes service that you can use to deploy containerized applications to the cloud. OKE supports two types of worker nodes:

- *Managed* nodes run on compute instances (either bare metal or virtual machine) in your tenancy. You are responsible for managing managed nodes, but you have the flexibility to configure them to meet your specific requirements. You are responsible for upgrading Kubernetes on managed nodes and for managing cluster capacity.
- *Virtual* nodes enable you to run Pods at scale without the operational overhead of upgrading the data plane infrastructure and managing the capacity of clusters.

OKE does not support clusters with mixed node pools. A virtual node pool and managed node pool cannot be created in the same cluster.

Topics:

About TimesTen Database for OKE

TimesTen and the TimesTen Operator are available for deployment in a pre-existing OKE cluster for managed nodes as a stack listing in Oracle Cloud Marketplace: **Oracle TimesTen In-Memory System of Record for Oracle Cloud Infrastructure Kubernetes Engine-x86** After launching the stack, you use a simple wizard interface to configure the stack along with any supporting resources, such as the pre-existing OKE and network resources.

The TimesTen listings in Marketplace support these billing options:

- *Universal Credits (UCM)*, where you are billed for the cost of the Oracle TimesTen In-Memory Database license (based on the use case). The cost of the VMs running the TimesTen node pool, in addition to the cost of the compute resources, is billed separately. The **Oracle TimesTen In-Memory System of Record for Oracle Cloud Infrastructure Kubernetes Engine-x86** stack listing uses this option and is covered in this guide.

> **Note**
>
> From this point forward, the stack listing is referred as **Oracle TimesTen In-Memory Database for Oracle Cloud Infrastructure Kubernetes Engine** (**TimesTen Database for OKE**).

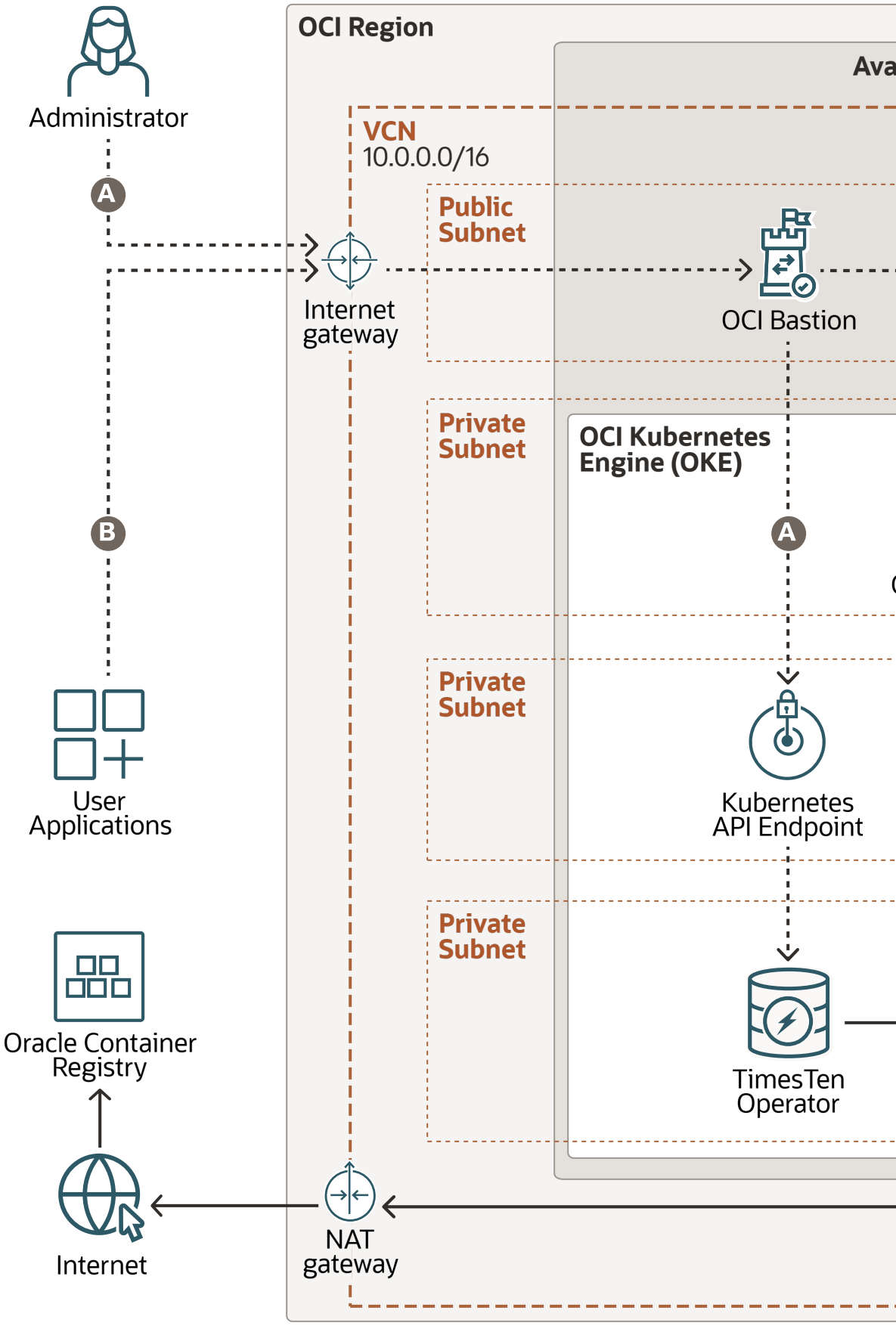
- *Bring Your Own License (BYOL)*, which allows you to reuse your existing on-premises Oracle TimesTen In-Memory Database license in Oracle Cloud Infrastructure (OCI). The **Oracle TimesTen In-Memory Database for Kubernetes - BYOL** container listing uses this option. See Using Oracle Cloud Marketplace to Obtain a TimesTen Container Image (BYOL).

TimesTen Database for OKE supports only the x86 architecture.

TimesTen Database for OKE supports only a *system of record* use case, where you use the **Oracle TimesTen In-Memory System of Record for Oracle Cloud Infrastructure Kubernetes Engine-x86** stack listing to set up standalone TimesTen databases in a OKE cluster. You can configure the TimesTen databases in either a replicated active standby pair configuration or a non-replicated configuration. See Deploy the TimesTen Operator and Create TimesTen Databases.

About the Components of TimesTen Database for OKE

Learn about the Oracle and OCI components that comprise TimesTen Database for OKE.



TimesTen

Oracle TimesTen In-Memory Database is the world's fastest OLTP database. It is a relational in-memory database with a rich feature set. TimesTen supports SQL, standard APIs, complete ACID properties, and highly available replication mechanisms. A TimesTen database resides entirely in physical memory and is persistent and recoverable. By managing data in memory and optimizing data structures and access algorithms, database operations run efficiently, achieving dramatic gains in responsiveness and throughput.

You can deploy TimesTen as a standalone database. You can use the TimesTen Operator to deploy, manage, and monitor your TimesTen databases in a Kubernetes environment.

OKE

Oracle Cloud Infrastructure Kubernetes Engine is a fully managed, scalable, and highly available service that you can use to deploy your containerized applications to the cloud. You specify the compute resources that your applications require, and OKE provisions them on OCI in an existing tenancy. OKE uses Kubernetes to automate the deployment, scaling, and management of containerized applications across clusters of hosts.

TimesTen Database for OKE creates a managed node pool on your pre-existing OKE cluster and deploys the TimesTen Operator on either an existing worker node or in one of the worker nodes from the new node pool.

TimesTen Operator

The TimesTen Kubernetes Operator has several key features to assist with managing TimesTen databases in a Kubernetes environment. A TimesTen database is modeled as a custom resource in the Kubernetes configuration file. The TimesTen Operator creates, manages, and monitors TimesTen Classic databases in both of the following topologies:

- Active standby pair replication scheme.
- Non-replicated configuration.

Marketplace

Oracle Cloud Marketplace is an online store that offers a catalog of listings from approved and registered publishers. You can use Marketplace to find an image, stack, container image, helm chart, and more, and deploy it on OCI.

TimesTen Database for OKE consists of a stack listing to deploy the TimesTen Operator. The TimesTen Operator enables you to deploy TimesTen databases as a system of record.

Resource Manager

Resource Manager is a service that uses Terraform to provision, update, and destroy a collection of related cloud resources as a single unit called a stack.

TimesTen Database for OKE uses Resource Manager to perform the following operations:

- If the Kubernetes API endpoint or the worker nodes of the specified OKE cluster are in private subnets, sets up a temporary Bastion.
- Creates a managed node pool in your OKE cluster with the specified number of worker nodes.
- Uses the provided Kubernetes Secret for Oracle Container Registry to pull the latest TimesTen container image into a temporary host in Resource Manager.

- Creates a temporary Podman container for the TimesTen container image and extracts the `ttcrd`, `ttoperator`, and `ttclassic` Helm charts.
- Installs the TimesTen Custom Resource Definition (CRD) using the `ttcrd` Helm chart.
- Creates a custom YAML file for the `ttoperator` Helm chart.
- Deploys the TimesTen Operator in the specified namespace using the custom YAML file and the `ttoperator` Helm chart.
- Deletes all temporary resources.

Oracle Container Registry

Oracle Container Registry is a repository of Docker containers for easy access to Oracle products.

TimesTen Database for OKE uses Oracle Container Registry to pull the latest TimesTen container image for the architecture specified. The stack uses the TimesTen container image to deploy the TimesTen Operator in your OKE cluster.

VCN and Subnets

A virtual cloud network (VCN) is a customizable, software-defined network that you set up in an OCI region. Like traditional data center networks, VCNs give you control over your network environment. A VCN can have multiple non-overlapping CIDR blocks that you can change after you create the VCN. You can segment a VCN into subnets, which can be scoped to a region or to an availability domain. Each subnet consists of a contiguous range of addresses that don't overlap with the other subnets in the VCN. You can change the size of a subnet after creation. A subnet can be public or private.

You can configure TimesTen Database for OKE to either use the same subnet for the Pods containing the TimesTen Operator and TimesTen databases as the worker nodes or use a different subnet.

Bastion

The Oracle Cloud Infrastructure Bastion provides restricted and time-limited secure access to resources that do not have public endpoints and that require strict resource access controls, such as Autonomous Transaction Processing (ATP), OKE, and any other resource that allows Secure Shell Protocol (SSH) access. With Bastion service, you can enable access to private hosts without deploying and maintaining a jump host. In addition, you gain improved security posture with identity-based permissions and a centralized, audited, and time-bound SSH session. Bastion removes the need for a public IP for bastion access, eliminating the hassle and potential attack surface when providing remote access.

To deploy the TimesTen Operator, TimesTen Database for OKE must be able to access your OKE cluster. If the Kubernetes API endpoint is in a private subnet, the stack creates a temporary Bastion plus all the underlying resources needed to access the Kubernetes API endpoint. The stack uses these temporary resources only to access the Kubernetes API endpoint for the sole purpose of deploying the TimesTen Operator in your OKE cluster. The stack does not access any other resource that may be available through the private subnet. The stack deletes all temporary resources after it completes all the tasks associated with deploying the TimesTen Operator in your OKE cluster.

Block Volume

With block storage volumes, you can create, attach, connect, and move storage volumes, and change volume performance to meet your storage, performance, and application requirements. After you attach and connect a volume to an instance, you can use the volume like a regular

hard drive. You can also disconnect a volume and attach it to another instance without losing data.

The TimesTen Operator creates a persistent volume claim (PVC) for each Pod containing a TimesTen database. Persistent volume claims must request a minimum of 50 gigabytes.

Compute

The Oracle Cloud Infrastructure Compute service enables you to provision and manage compute hosts in the cloud. You can launch compute instances with shapes that meet your resource requirements for CPU, memory, network bandwidth, and storage. After creating a compute instance, you can access it securely, restart it, attach and detach volumes, and terminate it when you no longer need it.

TimesTen Database for OKE enables you to select the shape of the compute instances for the worker nodes that hold the TimesTen databases.

Cloud Guard

You can use Oracle Cloud Guard to monitor and maintain the security of your resources in Oracle Cloud Infrastructure. Cloud Guard uses detector recipes that you can define to examine your resources for security weaknesses and to monitor operators and users for certain risky activities. When any misconfiguration or insecure activity is detected, Cloud Guard recommends corrective actions and assists with taking those actions, based on responder recipes that you can define.

Load Balancer

The Oracle Cloud Infrastructure Load Balancing service provides automated traffic distribution from a single entry point to multiple servers in the back end.

2

Deploy the TimesTen Operator

TimesTen Database for OKE deploys the TimesTen Operator in a specified namespace in an existing OKE cluster—the Pod for the TimesTen Operator can reside on any of the worker nodes already available in the cluster. It also creates a managed node pool on the same cluster for the later deployment of TimesTen databases. If there are no available node pools, the stack uses a worker node from the new node pool to deploy the TimesTen Operator.
To deploy the TimesTen Operator, the stack performs the following operations:

- If the Kubernetes API endpoint for OKE cluster is in a private subnet or you specified a private subnet for the worker nodes of the new node pool, creates temporary bastion and the necessary resources (a subnet, security rule, and bastion session) to access the OKE cluster.
- Creates a managed node pool in your OKE cluster with the specified number of worker nodes in the selected architecture.
- Using the provided Kubernetes Secret for Oracle Container Registry, pulls the latest TimesTen container image into a temporary host in Resource Manager.
- Creates a temporary container from the TimesTen container image and extracts the `ttcrd`, `ttoperator`, and `ttclassic` Helm charts.
- Installs the TimesTen Custom Resource Definition (CRD) using the `ttcrd` Helm chart.
- Creates a custom YAML file for the `ttoperator` Helm chart with the parameters for:
  - The TimesTen container image
  - The Secret for Oracle Container Registry
  - The architecture (amd64) of the worker nodes created by the stack
  - The environment label that ensures TimesTen databases are only created by the TimesTen Operator on worker nodes created by the stack
- Deploys the TimesTen Operator in the specified namespace using the custom YAML file and the `ttoperator` Helm chart.

  > **Note**
  >
  > The deployment process creates and deploys the service account, role, and rolebinding objects required to run the TimesTen Operator in the specified namespace.

- Deletes all temporary resources.

Topics:

Prerequisites to Deploy the TimesTen Operator

Before you create a stack to deploy the TimesTen Operator:

- You must have access to an OCI tenancy. The tenancy must be subscribed to one or more of the regions in which OKE is available. See Availability by Region.
- Your tenancy must have sufficient quota on:
  - *Compute instance quota*: The compute instance quota depends on the shape you select. Compute instance quota is determined by the compute cores and the amount of memory required for a particular shape. There should be enough quota for the specified number of worker nodes in the specified shape for the managed node pool for TimesTen databases. See Compute Limits.
  - *Block volume quota*: The TimesTen Operator creates a persistent volume claim (PVC) for each worker node containing a TimesTen database. Persistent volume claims must request a minimum of 50 gigabytes. See Setting Up Storage for Kubernetes Clusters.
  - *Bastion quota*: The stack creates a temporary bastion to access the OKE cluster, if necessary. There should be enough quota for at least one bastion. See Bastion Limits.
- Within your tenancy, there must already be an OKE cluster for managed node pools. See Creating Kubernetes Clusters Using Console Workflows.
  - The stack deploys the TimesTen Operator in a namespace at namespace scope. The stack configuration page requires that you provide a namespace for the TimesTen Operator and TimesTen databases. This namespace must be already configured in the OKE cluster before launching the stack. See About the TimesTen Operator in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* and Namespaces.
  - The stack configuration page requires that you provide a Kubernetes Secret with the credentials for Oracle Container Registry. For this, you must have access to an Oracle Container Registry account. To deploy the TimesTen Operator, the stack pulls the latest TimesTen container image from Oracle Container Registry using the provided Secret. See Example 2-1.

    You must agree to and accept the Oracle Standard Terms and Restrictions for downloading official container images of Oracle TimesTen In-Memory Database from Oracle Container Registry before launching the stack. To agree to and accept the Oracle Standard Terms and Restrictions, do the following:

    1. On the Oracle Container Registry homepage, click on **TimesTen**.
    2. On the **TimesTen Repositories** page, click **Continue** for the **timesten** repository.
    3. Read and understand the terms set by the **Oracle Standard Terms and Restrictions** pop-up, then click **Accept**.
- Within your tenancy, there must already be a compartment that contains the necessary network resources, such as:
  - VCN
  - Subnets
  - Internet gateway
  - NAT gateway
  - Service gateway
  - Route table

  - Security lists

  These resources must be configured in the region in which you want to deploy TimesTen databases. These resources can be the same as those used by the OKE cluster. See Network Resource Configuration for Cluster Creation and Deployment.

- Within your tenancy, there must already be a compartment to contain the stack. This compartment can be the same as the one associated with the OKE cluster.
- To create and manage the stack and OKE cluster, you must belong to one of the following:
  - The Administrators group of the tenancy.
  - A group to which a policy grants the appropriate Marketplace and OKE permissions. See Required IAM Policies to Deploy the TimesTen Operator.
- You must have a development host to access the OKE cluster. The host can be a virtual machine (VM) instance in the same tenancy as your cluster. Ensure that:
  - The host resides outside the OKE cluster.
  - The host is able to access and control the OKE cluster. For this, ensure that the `podman` (or `docker`) and `kubectl` command line tools are installed on the host. See the Podman and kubectl documentation.

    The stack uses Helm to deploy the TimesTen Operator and most of the examples in this guide use Helm. To be able to follow the examples, ensure that the `helm` command line tool is also installed. See the Helm documentation.

  See Setting Up Cluster Access.

Example 2-1 Creating a Kubernetes Secret to Oracle Container Registry

This example creates a Kubernetes Secret to Oracle Container Registry (OCR).

1. From your development host, log in to OCR,

   ```
   podman login container-registry.oracle.com
   ```

   a. When prompted for a user name, enter your username for OCR.

   b. When prompted for a password, enter your auth token for your OCR username.

2. Create a Kubernetes Secret in the namespace you designated for the TimesTen Operator with the credentials stored in your authentication file.

   ```
   kubectl create secret generic <secret-name> \
     --from-file=.dockerconfigjson=<path> \
     --type=kubernetes.io/dockerconfigjson \
     --namespace <namespace>
   ```

> **Note**
>
> The default path for the authentication file is `${XDG_RUNTIME_DIR}/containers/auth.json` on Linux and `$HOME/.config/containers/auth.json` on Windows or macOS.

Required IAM Policies to Deploy the TimesTen Operator

If you do not belong to the Administrators group of the tenancy, your group must have `manage`, `read`, or `use` access to certain resources. These resources include:

- The compartment in which you want to create the stack.
- The compartment in which the OKE cluster resides.
- The compartment in which the VCN resides.

The minimum required policies to deploy the TimesTen Operator using TimesTen System of Record for OKE are:

- General policies

  ```
  Allow group <group-name> to inspect all-resources in tenancy
  ```

- Policies for using Marketplace

  ```
  Allow group <group-name> to read marketplace-workrequests in compartment
  <stack-compartment-name>
  Allow group <group-name> to use marketplace-listings in compartment <stack-
  compartment-name>
  Allow group <group-name> to manage app-catalog-listings in compartment
  <stack-compartment-name>
  ```

- Policies for launching the stack

  ```
  Allow group <group-name> to manage orm-stacks in compartment <stack-
  compartment-name>
  Allow group <group-name> to manage orm-jobs in compartment <stack-
  compartment-name>
  ```

- Policies for the resources required by the stack
  - Policies for creating the node pool

    ```
    Allow group <group-name> to manage cluster-node-pools in compartment
    <cluster-compartment-name>
    Allow group <group-name> to manage instance-family in compartment
    <cluster-compartment-name>
    Allow group <group-name> to read cluster-work-requests in compartment
    <cluster-compartment-name>
    Allow group <group-name> to use vnics in compartment <cluster-
    compartment-name>
    Allow group <group-name> to use subnets in compartment <vcn-compartment-
    name>
    ```

> **Note**
>
> If the Kubernetes API endpoint is in a private subnet, use `manage` instead of `use` in the policy for the subnets of the compartment in which the VCN resides.

- Policies for deploying the TimesTen Operator

  ```
  Allow group <group-name> to manage clusters in compartment <cluster-compartment-name>
  ```

- Policies for using the Bastion service

  > **Note**
  >
  > The following policies are only necessary if the Kubernetes API endpoint is in a private subnet.

  ```
  Allow group <group-name> to manage security-lists in compartment <vcn-compartment-name>
  Allow group <group-name> to manage vcns in compartment <vcn-compartment-name>
  Allow group <group-name> to manage route-tables in compartment <vcn-compartment-name>
  Allow group <group-name> to manage dhcp-options in compartment <vcn-compartment-name>
  Allow group <group-name> to manage bastion-family in compartment <vcn-compartment-name>
  ```

Launch the Stack to Deploy the TimesTen Operator

To deploy the TimesTen Operator from the stack listing on Marketplace:

1. Sign in to your tenancy in Oracle Cloud Console (Console).
2. In the top-left corner of the Console, expand the Navigation menu and click **Marketplace**. Then, click **All Applications**.
3. Click on the **Oracle TimesTen In-Memory System of Record for Oracle Cloud Infrastructure Kubernetes Engine-x86** stack listing.

   To find the listing, you may use the filters and search bar as follows:

   a. In Filters, in the **Type** drop-down list, select **Stack**.
   b. In the **Publisher** drop-down list, select **Oracle**.
   c. In the **Price** drop-down list, select **Paid**.
   d. To narrow your search even further, in the search bar, type **TimesTen**.
4. In the **Version** drop-down list, select the version that matches the Kubernetes version of your OKE cluster.
5. In the **Compartment** drop-down list, select a compartment with the appropriate IAM policies.

   This is the compartment where the stack will be created. See Required IAM Policies to Deploy the TimesTen Operator.
6. Review and accept the **Oracle Standard Terms and Restrictions**.
7. Click **Launch Stack**.
8. (Optional) Provide a name for the stack.

9. (Optional) Provide a description for the stack.
10. Click **Next**.
11. In **OKE cluster configuration**, select the compartment of your OKE cluster.
12. Select your OKE cluster.
13. Specify the namespace for the TimesTen Operator and TimesTen databases.

    This namespace must already exist in the OKE cluster.
14. Specify the Kubernetes Secret with the credentials for Oracle Container Registry.

    The Secret must already exist in the OKE cluster.
15. In **Worker nodes configuration**, provide the name for the node pool to be created for TimesTen databases.

    Due to license restrictions, TimesTen databases can only be created in the worker nodes from this node pool.
16. Select the shape of the worker nodes.

    Since TimesTen is an in-memory database, select a shape that provides as much memory as required by the data and workloads intended for the TimesTen databases.
17. Specify the boot volume size for each worker node.

    The size of the boot volume should be enough to store all TimesTen-related files.
18. Specify the number of worker nodes.

    For high availability, it is recommended that each TimesTen database uses its own worker node. For example, an active standby pair replication scheme consists of two TimesTen databases, an active and standby. For this configuration, a minimum of two worker nodes is recommended.
19. In **Network and placement configuration**, provide the compartment of the VCN used by the OKE cluster.
20. Select the name of the VCN.
21. Select the name of the subnet used by any existing worker nodes in the OKE cluster.

    Ensure that you do not specify the subnet used for the Kubernetes API endpoint. Otherwise, deployment of the worker nodes of the new node pool fails.
22. Either upload the SSH public key file or paste the SSH public key for SSH access to the worker nodes.
23. In **Pod networking configuration**, select the subnet that the Pods for TimesTen databases should use.

    Ensure that you do not specify the subnet used for the Kubernetes API endpoint. Otherwise, deployment of the worker nodes of the new node pool fails.
24. Click **Next**.
25. Verify all the configuration variables.
26. Select the **Run apply** checkbox.

    This ensures that the required resources are provisioned immediately and starts the deployment of the TimesTen Operator and the creation of the node pool for TimesTen databases.
27. Click **Create**.

This operation is asynchronous. To monitor its progress, check the job details for the stack in Resource Manager. See Getting a Job's Details.

Test the TimesTen Operator

Once the apply job operation is complete for the stack, you can verify that the TimesTen Operator is up and running and in the specified namespace using the `helm test` command on your OKE cluster.

> **Note**
>
> To test the TimesTen Operator, the test Pod uses the `curl` command to access the readiness probe endpoint of the TimesTen Operator. If the TimesTen Operator self-reports that it is ready, the test succeeds.

1. Determine the release for the `ttoperator` Helm chart.

   ```
   helm list -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME    NAMESPACE    REVISION  UPDATED         STATUS
   CHART                   APP VERSION
   ttcrd   <namespace>  1         2025-05-20 ...  deployed
   ttcrd-2211350.1.0       22.1.1.35.0
   ttoper  <namespace>  1         2025-05-20 ...  deployed
   ttoperator-2211350.1.0  22.1.1.35.0
   ```

   The `ttoper` release of the `ttoperator` Helm chart is deployed in the specified namespace.

2. Test the `ttoper` release.

   ```
   helm test ttoper -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME: ttoper
   LAST DEPLOYED: Tue May 20 23:20:54 2025
   NAMESPACE: <namespace>
   STATUS: deployed
   REVISION: 1
   TEST SUITE:     ttoper-ttoperator-test
   Last Started:   Thu May 22 17:04:55 2025
   Last Completed: Thu May 22 17:05:00 2025
   Phase:          Succeeded
   NOTES:
   Version 2211320.1.0 of the ttoperator chart has been installed.

   This release is named "ttoper".

   To learn more about the release, try:
   ```

```
  $ helm status ttoper
  $ helm get all ttoper
  $ helm history ttoper

To rollback to a previous version of the chart, run:

  $ helm rollback ttoper <REVISION>
    - run 'helm history ttoper' for a list of revisions.

To test the operator, run:

  $ helm test ttoper
```

A successful test indicates the TimesTen Operator is running and properly operating in the specified namespace.

To check the worker nodes created by the stack, you can use the `kubectl get nodes` command.

```
kubectl get nodes --selector='license.timesten.oracle.com=1'
```

Output should include as many worker nodes as defined in the stack configuration page:

```
NAME          STATUS   ROLES   AGE   VERSION
10.0.10.101   Ready    node    5m    v1.33.1
10.0.10.39    Ready    node    5m    v1.33.1
```

> **Note**
>
> The `license.timesten.oracle.com=1` label is added to every worker node created by the stack. The TimesTen Operator uses this label to identify which worker nodes are licensed to run TimesTen databases.

Alternatively, you can use the OCI Console to check the details of the new node pool and worker nodes, see Getting a Managed Node Pool's Details.

3

Create TimesTen Databases

TimesTen Database for OKE provides the ability to use TimesTen databases as a system of record in either a non-replicated environment or in an active standby pair replication scheme. The active standby pair replication scheme configuration is recommended for highest availability. An active standby pair replication scheme includes an active TimesTen database and a standby TimesTen database.
When replication is configured, a replication agent is started for each TimesTen database. The active TimesTen database uses the replication agent to send updates to the standby database, and the standby TimesTen database uses its own replication agent to receive them. Each of these connections is implemented as a separate thread running inside the replication agent process. Replication agents communicate through TCP/IP stream sockets.

For maximum performance, the replication agent detects updates to the active TimesTen database by monitoring the existing transaction log. It then sends these updates to the standby TimesTen database in batches, if possible. Only committed transactions are replicated. On the standby TimesTen database, the replication agent updates the database through an efficient internal interface that avoids the overhead of the SQL layer.

Configuration metadata lets you define the attributes of your TimesTen database and how that database is to interact with other applications and components. The TimesTen Operator supports several metadata files that contain the configuration metadata. Each metadata file has a specific name and purpose. For example:

File Name	Description
`adminUser`	Defines a database user in the TimesTen database and assigns `ADMIN` privileges to it.
`db.ini`	Defines the connection attributes of the TimesTen database.
`schema.sql`	Defines database objects, such as tables, sequences, and users.
`testUser`	Defines a database user in the TimesTen database and assigns `CONNECT` privileges to it. This user is required for the `helm test` command to work if the `ttclassic` Helm chart is used to create the TimesTen databases.

There are other metadata files available that enable you to configure other attributes and metadata for the TimesTen database, but they are not required. These configuration options include:

- Defining operations to be performed after the replication scheme has been configured
- Configuring TLS for client/server communication to the TimesTen database
- Configuring an Oracle Wallet for replication operations between TimesTen databases

See About Configuration Metadata Details in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

These metadata files can then be used on one or more Kubernetes facilities—such as a ConfigMap, Secret, or init container—to configure TimesTen databases. The TimesTen

Operator creates a projected volume called `tt-config`. This `tt-config` volume contains the contents of all the ConfigMaps and Secrets specified in the `dbConfigMap` and `dbSecret` fields of your TimesTenClassic object. This volume is mounted as `/ttconfig` in the TimesTen containers. See Populate the /ttconfig Directory in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

The TimesTen Operator uses the TimesTenClassic object type to provide the metadata and attributes to define and create TimesTen Classic databases in the specified namespace of your OKE cluster. The metadata and attributes are applicable to replicated and non-replicated configurations. See About Defining TimesTenClassic Objects and About the TimesTenClassic Object Type in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

The following table consists of a typical sequence of tasks to create and use TimesTen databases in a Kubernetes cluster.

Task	Description	More information
Create the metadata files.	Create the metadata files with the configuration attributes and metadata for the TimesTen databases, such as a database user with `ADMIN` privileges, the connection attributes for the TimesTen databases and the databases objects to be created in the databases.	About Configuration Metadata Details in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*
Populate the `/ttconfig` directory.	Create a ConfigMap, Secret or init container to incorparate the metadata files into the TimesTen containers.	Populate the /ttconfig Directory in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*
Create a TimesTenClassic object.	Create a TimesTenClassic object to provide the configuration attributes and metadata to define and create the TimesTen databases in your namespace in a replicated or non-replicated configuration.	Create Replicated TimesTen Classic Databases in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* or Create Non-Replicated TimesTen Classic Databases in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*
Perform operations on the TimesTen databases	Connect, manage, and monitor your TimesTen databases.	Use TimesTen Databases in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* Manage and Monitor TimesTen Classic Databases in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*

Some of the examples described in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* assume that you know the name and location of the TimesTen container image. You can retrieve this information from the description of the Pod of the TimesTen Operator in your specified namespace.

```
$ kubectl get pods -n <namespace>
NAME                              READY   STATUS    RESTARTS   AGE
timesten-operator-<unique-id>     1/1     Running   0          11m

$ kubectl describe pod timesten-operator-<unique-id> -n <namespace> | grep
Image:
   Image: container-registry.oracle.com/timesten/timesten:22.1.1.35.0
```

These examples also assume you know the name of the Secret with the credentials to Oracle Container Registry. Use the Secret specified during that stack configuration. Also, note that the `oci-bv` storage class is the default for OKE clusters.

Using Helm to Create the TimesTen Databases

If you want to create TimesTen databases using the `ttclassic` Helm chart included with the TimesTen Operator, see The ttclassic Chart and Create TimesTen Databases and Test TimesTen in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

You can create a local copy of the `ttclassic` chart from the `/timesten/operator/helm` directory in the Pod of the TimesTen Operator in the specified namespace.

```
kubectl cp timesten-operator-<unique-id>:/timesten/operator/helm/ttclassic
ttclassic -n <namespace>
```

4

Upgrade the TimesTen Operator and TimesTen Databases

New TimesTen releases may include bug fixes, security fixes, or new features for the TimesTen databases, or new versions of the TimesTen CRDs and TimesTen Operator. Every new TimesTen release is made available as a container image in Oracle Container Registry. The changes between TimesTen releases are described in Changes in This Release in *Oracle TimesTen In-Memory Database Release Notes*.
To upgrade the TimesTen Operator and TimesTen databases, you first have to determine which TimesTen release your TimesTen Operator and TimesTen databases are using. See Example 4-1 and Example 4-2.

Example 4-1 Determining the TimesTen Release of the TimesTen Operator

This example requests information on the Pod of the TimesTen Operator to determine the TimesTen release of the TimesTen container image.

```
$ kubectl get pods -n <namespace>
NAME                              READY   STATUS    RESTARTS   AGE
timesten-db-0                      3/3     Running   0          3d
timesten-db-1                      2/3     Running   0          3d
timesten-operator-<unique-id>      1/1     Running   0          3d

$ kubectl describe pod timesten-operator-<unique-id> -n <namespace> | grep
Image:
   Image: container-registry.oracle.com/timesten/timesten:22.1.1.35.0
```

Example 4-2 Determining the TimesTen Release of the TimesTen Databases

This example requests information on the `timesten-db` TimesTenClassic object to determine the TimesTen release of the TimesTen container image.

```
$ kubectl describe ttc timesten-db -n <namespace> | grep Image:
    Image:  container-registry.oracle.com/timesten/timesten:22.1.1.35.0
...
```

Once you determined that a new TimesTen release is available in Oracle Container Registry, follow the procedures described below to upgrade the TimesTen CRD, the TimesTen Operator, and the TimesTen Classic databases.

> **Note**
>
> All examples and procedures in this chapter assume that during the stack configuration you specified a namespace other than `default` for the deployment of the TimesTen Operator.

Topics:

Obtain the Helm Charts for Upgrade

Since the stack used Helm and the Helm charts included in a TimesTen container image to deploy the TimesTen Operator, you can use the Helm charts from a newer version of the TimesTen container image to upgrade the TimesTen Operator and TimesTen databases.

To obtain the Helm charts from a newer version of the TimesTen container image, do the following:

1. On your development host, use `docker login` to log in to the Oracle Container Repository (OCR).

   You should use the same credentials you used to create the Kubernetes Secret for the stack configuration. The OCR user used for that Secret has already accepted the **Standards Terms and Restrictions** for the official container images for TimesTen.

   ```
   podman login container-registry.oracle.com
   ```

2. Create a temporary container for the version of TimesTen container image you wish to use for the upgrade.

   ```
   podman create --name <temp-container> container-registry.oracle.com/
   timesten/timesten:22.1.1.36.0
   ```

   > **Note**
   >
   > This example assumes that the current version of TimesTen used by the TimesTen Operator and TimesTen databases is `22.1.1.35.0` and the version for the upgrade is `22.1.1.36.0`.

3. Copy the Helm charts for the upgrade to a local directory.

   ```
   podman cp <temp-container>:/timesten/operator/helm .
   ```

4. Delete the temporary container.

   ```
   podman rm <temp-container>
   ```

You have successfully acquired the Helm charts required to upgrade the TimesTen version of the TimesTen Operator and TimesTen databases.

Upgrade the TimesTen CRD

Since the stack used Helm and the Helm charts included in a TimesTen container image to deploy the TimesTen Operator, you can use the Helm charts from a newer version of the TimesTen container image to upgrade the TimesTen CRD for the TimesTenClassic object type.

To upgrade the TimesTen CRD, do the following:

1. On your development host, go to the directory where you stored the newer version of the Helm charts, see Obtain the Helm Charts for Upgrade.
2. Confirm the release for the `ttcrd` Helm chart.

   ```
   helm list -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME         NAMESPACE    REVISION  UPDATED         STATUS
   CHART                    APP VERSION
   timesten-db  <namespace>  1         2025-06-05 ...  deployed
   ttclassic-2211350.1.0   22.1.1.35.0
   ttcrd        <namespace>  1         2025-06-05 ...  deployed
   ttcrd-2211350.1.0       22.1.1.35.0
   ttoper       <namespace>  1         2025-06-05 ...  deployed
   ttoperator-2211350.1.0  22.1.1.35.0
   ```

   > **Note**
   >
   > This example assumes that the current version of TimesTen used by the `ttcrd` release is `22.1.1.35.0` and the version for the upgrade is `22.1.1.36.0`.

3. Upgrade the `ttcrd` release.

   ```
   helm upgrade ttcrd -n <namespace> helm/ttcrd
   ```

   Output should be similar to the following:

   ```
   Release "ttcrd" has been upgraded. Happy Helming!
   NAME: ttcrd
   LAST DEPLOYED: Thu Jun  5 17:28:39 2025
   NAMESPACE: <namespace>
   STATUS: deployed
   REVISION: 2
   TEST SUITE: None
   NOTES:
   Version 2211360.1.0 the ttcrd chart has been installed.

   This release is named "ttcrd".

   To learn more about the release, try:

     $ helm status ttcrd
   ```

```
  $ helm get all ttcrd
  $ helm history ttcrd

To rollback to a previous version of the chart, run:

  $ helm rollback ttcrd <REVISION>
    - run 'helm history ttcrd' for a list of revisions.
```

4. Confirm the release for the `ttcrd` Helm chart.

   ```
   helm list -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME         NAMESPACE    REVISION  UPDATED         STATUS
   CHART                     APP VERSION
   timesten-db  <namespace>  1         2025-06-05 ...  deployed
   ttclassic-2211350.1.0   22.1.1.35.0
   ttcrd        <namespace>  2         2025-06-05 ...  deployed
   ttcrd-2211360.1.0       22.1.1.36.0
   ttoper       <namespace>  1         2025-06-05 ...  deployed
   ttoperator-2211350.1.0  22.1.1.35.0
   ```

You have successfully upgraded the TimesTen CRD for the TimesTenClassic object type.

Upgrade the TimesTen Operator

Since the stack used Helm and the Helm charts included in a TimesTen container image to deploy the TimesTen Operator, you can use the Helm charts from a newer version of the TimesTen container image to upgrade the TimesTen Operator.

To upgrade the TimesTen Operator, do the following:

1. On your development host, go to the directory where you stored the newer version of the Helm charts, see Obtain the Helm Charts for Upgrade.
2. Create a YAML file where you specify the TimesTen container image from Oracle Container Registry to be used by the TimesTen Operator for the upgrade.

   ```
   image:
     repository: container-registry.oracle.com/timesten/timesten
     tag: "22.1.1.36.0"
   ```

   > **Note**
   >
   > This example assumes that the current version of TimesTen used by the `ttoper` release is `22.1.1.35.0` and the version for the upgrade is `22.1.1.36.0`.

3. Upgrade the `ttoper` release.

Ensure that you include the `--reuse-values` option, so that the values from the current `ttoper` release are used for the upgrade and only the values from the YAML file you just created are overridden.

```
helm upgrade ttoper -n <namespace> \
  -f <upgrade-file>.yaml helm/ttoperator --reuse-values
```

Output should be similar to:

```
Release "ttoper" has been upgraded. Happy Helming!
NAME: ttoper
LAST DEPLOYED: Thu Jun  5 18:27:38 2025
NAMESPACE: <namespace>
STATUS: deployed
REVISION: 2
NOTES:
Version 2211360.1.0 of the ttoperator chart has been installed.

This release is named "ttoper".

To learn more about the release, try:

  $ helm status ttoper
  $ helm get all ttoper
  $ helm history ttoper

To rollback to a previous version of the chart, run:

  $ helm rollback ttoper <REVISION>
    - run 'helm history ttoper' for a list of revisions.

To test the operator, run:

  $ helm test ttoper
```

4. Test the TimesTen Operator.

> ⚠ **Caution**
>
> Ensure you wait a couple minutes for the upgrade to complete before performing the test. Otherwise, the test may fail. If this is the case, you need to delete the Pod created by the test before you are able to run the test again.

```
helm test ttoper -n <namespace>
```

Output should be similar to:

```
NAME: ttoper
LAST DEPLOYED: Thu Jun  5 18:41:14 2025
NAMESPACE: <namespace>
STATUS: deployed
REVISION: 2
```

```
TEST SUITE:     ttoper-ttoperator-test
Last Started:   Thu Jun  5 18:42:44 2025
Last Completed: Thu Jun  5 18:44:11 2025
Phase:          Succeeded
NOTES:
Version 2211360.1.0 of the ttoperator chart has been installed.

This release is named "ttoper".

To learn more about the release, try:

  $ helm status ttoper
  $ helm get all ttoper
  $ helm history ttoper

To rollback to a previous version of the chart, run:

  $ helm rollback ttoper <REVISION>
    - run 'helm history ttoper' for a list of revisions.

To test the operator, run:

  $ helm test ttoper
```

You can further verify that the TimesTen Operator is using the new version of the TimesTen container image by running the following command:

```
kubectl get deployment timesten-operator -o yaml -n <namespace> | grep
image:
```

Output should be similar to:

```
  image: container-registry.oracle.com/timesten/timesten:22.1.1.36.0
```

You have successfully upgraded the TimesTen Operator.

Upgrade the TimesTen Databases

Assuming you used the Helm charts included in the TimesTen container image to create the TimesTen databases, you can use the Helm charts from a newer version of the TimesTen container image to upgrade the TimesTen databases.
The procedure to upgrade the TimesTen databases slightly differs depending if the databases are in a replicated or non-replicated scheme.

- Upgrade Replicated TimesTen Databases
- Upgrade Non-Replicated TimesTen Databases

Upgrade Replicated TimesTen Databases

Before upgrading replicated TimesTen databases, verify that their state is `Normal`.

```
$ kubectl get ttc -n <namespace>
NAME          STATE    ACTIVE          AGE
timesten-db   Normal   timesten-db-0   6h
```

If the state of the TimesTenClassic object differs from `Normal`, see About the High Level State of TimesTenClassic Objects in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* for more information.

To upgrade replicated TimesTen databases, do the following:

1. On your development host, go to the directory where you stored the newer version of the Helm charts, see Obtain the Helm Charts for Upgrade.
2. Create a YAML file where you specify the TimesTen container image from Oracle Container Registry to be used by the TimesTen databases for the upgrade.

   ```
   image:
     repository: container-registry.oracle.com/timesten/timesten
     tag: "22.1.1.36.0"
   ```

   > **Note**
   >
   > This example assumes that the current version of TimesTen used by the release of the `ttclassic` chart is `22.1.1.35.0` and the version for the upgrade is `22.1.1.36.0`.

3. Confirm the release for the `ttclassic` Helm chart.

   ```
   helm list -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME         NAMESPACE    REVISION  UPDATED         STATUS
   CHART                     APP VERSION
   timesten-db  <namespace>  1         2025-06-05 ...  deployed
   ttclassic-2211350.1.0   22.1.1.35.0
   ttcrd        <namespace>  2         2025-06-05 ...  deployed
   ttcrd-2211360.1.0       22.1.1.36.0
   ttoper       <namespace>  2         2025-06-05 ...  deployed
   ttoperator-2211360.1.0  22.1.1.36.0
   ```

   > **Note**
   >
   > In this example, the name of the release for the `ttclassic` chart is `timesten-db`. Your own setup should reflect the name you assigned to the release, since it is an user-defined name.

4. Upgrade the release for the `ttclassic` chart.

   Ensure that you include the `--reuse-values` option, so that the values from the current release for the `ttclassic` chart are used for the upgrade and only the values from the YAML file you just created are overridden.

   ```
   helm upgrade timesten-db -n <namespace> -f <upgrade-file>.yaml helm/
   ttclassic --reuse-values
   ```

   Output should be similar to:

   ```
   Release "timesten-db" has been upgraded. Happy Helming!
   NAME: timesten-db
   LAST DEPLOYED: Thu Jun  5 19:14:22 2025
   NAMESPACE: <namespace>
   STATUS: deployed
   REVISION: 2
   NOTES:
   Version 2211360.1.0 of the ttclassic chart has been installed.

   This release is named "timesten-db".

   To learn more about the release, try:

     $ helm status timesten-db
     $ helm get all timesten-db
     $ helm history timesten-db

   To rollback to a previous version of the chart, run:

     $ helm rollback timesten-db <REVISION>
       - run 'helm history timesten-db' for a list of revisions.
   ```

   The operator automatically upgrades the TimesTen databases in the active standby pair replication scheme. This operation is asynchronous. To monitor its progress, use the `kubectl get events -w -n <namespace>` command. The upgrade is complete once the TimesTenClassic object is back to `Normal` state.

5. Verify the state of the TimesTenClassic object.

   ```
   kubectl get ttc -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME          STATE    ACTIVE          AGE
   timesten-db   Normal   timesten-db-1   6h
   ```

> **Note**
>
> Due to the upgrade, which TimesTen database is the active and which is the standby changed. In this example, previous to the upgrade, the active was `timesten-db-0` and the standby was `timesten-db-1`. After the upgrade, `timesten-db-1` is now the active and `timesten-db-0` is the standby.

6. Verify that TimesTen container image has been upgraded for the TimesTenClassic object.

   ```
   kubectl describe ttc timesten-db -n <namespace> | grep Image:
   ```

   Output should be similar to:

   ```
     Image:  container-registry.oracle.com/timesten/timesten:22.1.1.36.0
   ...
   ```

You have successfully upgraded the replicated TimesTen databases.

Upgrade Non-Replicated TimesTen Databases

Before upgrading non-replicated TimesTen databases, verify that their state is `AllReplicasReady`.

```
$ kubectl get ttc -n <namespace>
NAME          STATE              ACTIVE   AGE
timesten-db   AllReplicasReady   N/A      1h
```

If the state of the TimesTenClassic object differs from `AllReplicasReady`, see About the High Level State of TimesTenClassic Objects in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide* for more information.

To upgrade non-replicated TimesTen databases, do the following:

1. On your development host, go to the directory where you stored the newer version of the Helm charts, see Obtain the Helm Charts for Upgrade.
2. Create a YAML file where you specify the TimesTen container image from Oracle Container Registry to be used by the TimesTen databases for the upgrade.

   Use the `rollingUpdatePartition` variable to determine which Pods of the non-replicated TimesTen databases to upgrade. The default value is`0`, which upgrades all Pods. See The ttclassic Chart in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

   ```
   image:
     repository: container-registry.oracle.com/timesten/timesten
     tag: "22.1.1.36.0"
   rollingUpdatePartition: 0
   ```

> **Note**
>
> This example assumes that the current version of TimesTen used by the release of the `ttclassic` Helm chart is `22.1.1.35.0` and the version for the upgrade is `22.1.1.36.0`.

3. Confirm the release for the `ttclassic` Helm chart.

   ```
   helm list -n <namespace>
   ```

   Output should be similar to:

   ```
   NAME         NAMESPACE    REVISION  UPDATED         STATUS
   CHART                     APP VERSION
   timesten-db  <namespace>  1         2025-06-05 ...  deployed
   ttclassic-2211350.1.0   22.1.1.35.0
   ttcrd        <namespace>  2         2025-06-05 ...  deployed
   ttcrd-2211360.1.0       22.1.1.36.0
   ttoper       <namespace>  2         2025-06-05 ...  deployed
   ttoperator-2211360.1.0  22.1.1.36.0
   ```

   > **Note**
   >
   > In this example, the name of the release for the `ttclassic` chart is `timesten-db`. Your own setup should reflect the name you assigned to the release, since it is an user-defined name.

4. Upgrade the release for the `ttclassic` chart.

   Ensure that you include the `--reuse-values` option, so that the values from the current release for the `ttclassic` chart are used for the upgrade and only the values from the YAML file you just created are overridden.

   ```
   helm updated timesten-db -n <namespace> -f <upgrade-file>.yaml helm/
   ttclassic --reuse-values
   ```

   Output should be similar to:

   ```
   Release "timesten-db" has been upgraded. Happy Helming!
   NAME: timesten-db
   LAST DEPLOYED: Thu Jun  5 19:45:03 2025
   NAMESPACE: <namespace>
   STATUS: deployed
   REVISION: 2
   NOTES:
   Version 2211360.1.0 of the ttclassic chart has been installed.

   This release is named "timesten-db".

   To learn more about the release, try:
   ```

```
  $ helm status timesten-db
  $ helm get all timesten-db
  $ helm history timesten-db

To rollback to a previous version of the chart, run:

  $ helm rollback timesten-db <REVISION>
    - run 'helm history timesten-db' for a list of revisions.
```

The operator automatically upgrades the TimesTen databases in the Pods specified by the `rollingUpdatePartition` variable. This operation is asynchronous. To monitor its progress, use the `kubectl get events -w -n <namespace>` command. The upgrade is complete once the TimesTenClassic object is back to `AllReplicasReady` state.

5. Verify the state of the TimesTenClassic object.

```
kubectl get ttc -n <namespace>
```

Output should be similar to:

```
NAME          STATE              ACTIVE   AGE
timesten-db   AllReplicasReady   N/A      1h
```

6. Verify that TimesTen container image has been upgraded for the TimesTenClassic object.

```
kubectl describe ttc timesten-db -n <namespace> | grep Image:
```

Output should be similar to:

```
    Image:  container-registry.oracle.com/timesten/timesten:22.1.1.36.0
...
```

You have successfully upgraded the non-replicated TimesTen databases.

5

Delete the TimesTen Operator and Node Pool

You can delete the TimesTen Operator by destroying the stack you created for TimesTen Database for OKE. Destroying the stack will remove all the resources that the stack created including the managed node pool and worker nodes created for TimesTen databases. However, this operation does not remove the user-provided Secret with the Oracle Container Registry credentials, any user-created ConfigMaps and Secrets used to create the TimesTenClassic objects, or the TimesTenClassic objects. It is highly recommended that you, at least, delete any TimesTenClassic objects you created before destroying the stack, see Clean Up or Delete TimesTen Databases in *Oracle TimesTen In-Memory Database Kubernetes Operator User's Guide*.

If the stack deployed the TimesTen Operator on a worker node outside of the node pool the stack created for TimesTen databases, the destroy stack operation only removes the Pod for the TimesTen Operator and any resources the stack originally added to the OKE cluster configuration to deploy the TimesTen Operator, such as the TimesTenClassic CRD, the role and service account for the TimesTen Operator.

> **Note**
>
> Deleting the stack and destroying the stack are separate operations and perform different actions:
>
> - **Destroy stack:** Removes all resources deployed by the stack, but the stack remains available in Resource Manager and you can recreate all thsese resources by rerunning the apply job.
> - **Delete stack:** Removes the stack from Resource Manager, but any associated resources persist. If the stack has not been destroyed first, you will need to remove the all resources deployed by the stack by different means.
>
> To ensure that all resources are removed and the stack deleted from Resource Manager, destroy and then delete the stack.

Topics:

Destroy the Stack

To destroy the stack, which deletes the TimesTen Operator and node pool:

1. Sign in to your tenancy in Oracle Cloud Console (Console).
2. In the top-left corner of the Console, expand the Navigation menu and click **Developer Services**. Then, under **Resource Manager**, click **Stacks**.
3. Click the name of the stack you want to destroy.
4. Click **Destroy**.

All resources deployed by the stack are removed immediately. This action cannot be reversed.

5. Click **Destroy**.

This operation is asynchronous. To monitor its progress, check the job details for the stack in Resource Manager. See Getting a Job's Details.

Delete the Stack

To delete the stack, which removes the stack from Resource Manager:

1. Sign in to your tenancy in Oracle Cloud Console (Console).
2. In the top-left corner of the Console, expand the Navigation menu and click **Developer Services**. Then, under **Resource Manager**, click **Stacks**.
3. Click the name of the stack you want to delete.
4. Click **More actions**, then click **Delete stack**.
5. Select the checkbox to indicate that you understand what happens if you delete the stack before running a destroy job.

   All resources deployed by the stack persist after the deletion of the stack.
6. Click **Delete**.

This operation is asynchronous. To monitor its progress, check the job details for the stack in Resource Manager. See Getting a Job's Details.