Best Practices for Large Deployments in the Oracle Service Cloud

ORACLE WHITE PAPER | JULY 2016

Contents

Introduction

The Oracle Service Cloud is powerful and easy to use. This is especially true when customers implement best practices for deploying their Service Cloud presence, learn how to manage their data, and understand limits of the various features of the Service Cloud. The guidance in this document applies to all Service Cloud customers, but it is especially helpful for customers with globally-diverse customer bases, processing hundreds of thousands of customer engagements daily, who must comply with one or more regulatory requirements, or who are targeting internal and external customers.

Intended Audience

This publication is for consideration by anyone involved in designing a successful Service Cloud deployment and anyone responsible for administering the Oracle Service Cloud environment.

Deployment Strategies

There are two basic strategies for deploying an Oracle Service Cloud presence: single-instance and distributed multiple instances. Numerous factors will influence the deployment approach that is best for your specific needs. Some factors to consider are:

- The natural segmentation of products, regions, or organization
- Anticipated volume of incident creation
- Anticipated volume of daily incident activity
- Geography of customers
- Number and geography of agents
- Regional data privacy or sovereignty laws
- Compliance requirements
- Sensitive data (PII, ePHI, PAN) processing requirements
- Risk mitigation strategy
- Other specific use cases for your business

As you can see, numerous considerations will affect the design of your organization's Service Cloud deployment, and Oracle Service Cloud can adapt to specific needs.

Single-Instance Deployment

In a single-instance deployment, there is one solitary database into which all data – incidents, contacts, answers, tasks, rules, etc. – will reside for agents and customers globally.

Benefits

The notable benefits of a single-instance deployment include:

- Ease of configuration

- Simplified reporting
- Single knowledgebase
- Centralized administration
- Manage a single, consolidated environment
- Consolidated code management
- Central point for application integration
- Customer data is accessible by all agents

Challenges

A single-instance deployment comes with many challenges for large customer deployments that may not be obvious at first, but that can eventually surface. These include:

- Network latency may impact distant users
- Analytics reports may require longer run times as data volume increases
- Archiving may lag behind incident creation in certain high volume creation situations
- Maintenance activities may take longer to run
- Maintenance activities may lead to resource contention that may be visible to agents and customers
- A larger database will make troubleshooting and upgrades difficult
- Many interfaces (languages, brands) associated to a single instance could impact overall performance
- Optlists (lists of products, categories, or accounts) tend to be large which can degrade performance

Recommendations for Single-Instance Deployment

A single-instance deployment is best suited for customers for which one or more of the following attributes applies:

- Local user base
- Centralized agents
- Low incident creation volume
- Limited administrative resources
- Agents or users who may be accustomed to network latency
- Low number of interfaces required

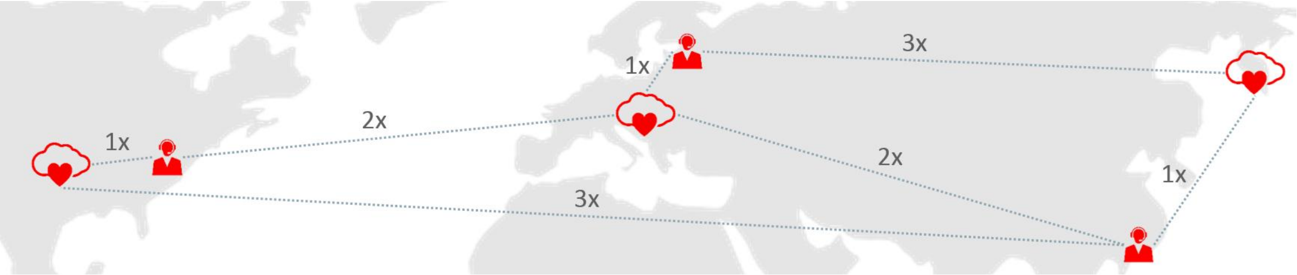
Examples of Single-Instance Deployments

Some different examples where a single instance is common are:

- A customer who serves users in a national region but not global
- A customer with a base of 2,000 agents who are in the same geographical region
- A customer who creates 30,000 incidents per day
- A customer with an administrative team of just a few people
- A customer serving a geographic region where low latency is the norm

Distributed Deployment

A distributed deployment is where multiple Service Cloud instances are created to be more geographically aligned with your organizations' agents and customers. For example, if a significant agent presence in more than one location – say United States, Europe, and Asia – there could be three separate instances, one for each geographically significant area.



Example of the effect latency may have on distributed users. (The numbers represent a multiple of latency.)

Benefits

For distributed deployments, benefits include:

- Generally better performance due to reduced network latency
- Better analytics performance due to distributed data sets
- Better opportunity for archiving to keep pace with incident creation
- More efficient maintenance activities that are less likely to be visible to agents and customers
- Larger natural maintenance windows than with follow-the-sun operations on a single instance
- Improved ability to roll out process changes and upgrades
- Improved ability to troubleshoot problems due to a smaller instance footprint
- Easier upgrades
- Improved performance of search across multiple knowledgebases that ought not be consolidated
- Usually better performance when many interfaces (languages, brands, etc.) are distributed across multiple instances
- Usually Optlists can be smaller since the data comprising them is distributed

Challenges

Common challenges for distributed deployments include:

- A potentially more complex configuration due to having multiple instances
- Reporting across multiple instances is manual unless using a third-party BI tool (recommended)
- Multiple knowledge bases to maintain
- Decentralized administration
- Multiple environments to manage
- Distributed code management
- Multiple points for application integration
- Customer data will be segmented and possibly redundant

Recommendations for Distributed Deployment

A distributed deployment is best for customers having one or more of the following attributes:

- A large and geographically-distributed user base
- A large and geographically-distributed agents base
- High incident creation volume
- Distributed administrative resources
- Agents or users are not accustomed to high latency
- Competing or conflicting regulatory requirements
- Preference for segregating different data types (i.e. ePHI vs. non-ePHI)
- Logical divides of knowledge (i.e. KB for one region is necessarily different from KB for another region)
- Twenty or more languages or brands

Examples of Distributed Deployments

Different examples where a distributed environment is common are:

- A customer who services users distributed globally
- A customer with a base of 7,000 agents geographically distributed across multiple regions
- A customer who regularly creates 70,000 incidents per day
- A customer expecting low network latency for their entire organization
- A customer with a discreet group of users for which the customer expects to properly manage ePHI data.
- A customer with some brands in some geographical areas and other brands in other areas.

Recommendations

Consider the various factors that drive the type of deployment right for your organization. Take into account the benefits and risks associated with each type. Oracle Service Cloud can adapt to the unique characteristics of your organization. Taking the time to plan your deployment in advance gives you the opportunity to deliver the quality customer service your customers deserve.

Regional Data Privacy and Sovereignty Laws

The global information age is getting complex regarding the storage and processing of data, and it is getting noticed by world governments. Deployment Strategies are likely to be impacted by both existing regulations and new regulations that are springing up across the globe. For the most part, these regulations affect where PII data (personally identifiable information data) can be stored and who has access to it. Some of the most prolific laws that govern cloud-based data today are listed below.

Regulations that May Affect Deployment

Here are some regulations and regulatory frameworks that may affect storage and processing of PII data:

- U.S. Patriot Act
- Canada Anti-Terrorism Act (ATA)

- U.K. Regulation of Investigatory Act (RIPA)
- France Law Nos 2001-1062 and 2006-64
- EU Directive 95/46/EC (1995)
- EU General Data Protection Regulation (will replace EU Directive 95/46/EC)
- U.S. Privacy Shield (this framework will soon replace the U.S.-EU Safe Harbor Framework)
- APEC Privacy Framework
- Dutch Data Protection Act (DPA)
- Italian Data Protection Code
- U.K. Data Protection Act (2000)
- Russia 242-FZ
- Philippines Data Privacy Act of 2012 (RA-10173)

Many of these regulations dictate what kind of data must reside in which location or region. Other laws affect how data must be kept private, shared, or otherwise protected and managed. Regardless of the requirements and regulations to which you must abide, Oracle Service Cloud is prepared to help you comply. As you plan your deployment in the Oracle Service Cloud, recognize that these and other laws and regulations in those global regions relevant to you and your customers could have an impact on your plans.

This is not legal advice. For the laws affecting your data and the regulations applicable to you, consult your legal department for guidance.

Data Management Guidelines

To maximize success in the Oracle Service Cloud, proper data management is crucial. Here, we give you the information you need about why you should manage your Service Cloud data properly, lay out the importance of your incident data lifecycle, explain the value of archiving incidents, and share best practices to manage other data in your Service Cloud instance. We will also discuss what can happen if you do not manage your data.

Benefits of Keeping Service Cloud Databases Clutter-Free

Generally speaking, it is a best practice to establish a data management strategy, and this includes data stored within the Oracle Service Cloud. Successful customers actively manage data so it does not get out of control and have impacts on performance, especially performance that is visible to customers or agents. Each Service Cloud instance has its own database, and the size of this database affects a lot of different things. The database size will impact the speed of various operations, like:

- Refreshing test data in a test site
- Adding custom fields to large tables in the database
- Preparations before an upgrade
- Duplicating (cloning) a database for Tech Support or for your own use
- Recovering from a catastrophic failure

As you can see, some important lifecycle operations are impacted by the size of your database. On a periodic basis, you may want to refresh your test site with more current data from your production site. If you add a custom field to an existing table or add fields to a custom object, the Service Cloud seamless deployment process requires an amount of time related to the size of that table/object. When preparing for an upgrade, time is needed to clone your site and database prior to user acceptance testing. When you engage Tech Support, depending on the problem, they may need to create a clone to work with. If a recovery must be performed for any reason, you want that to be carried out as swiftly as possible. If you manage your Service Cloud data well, these operations will be much more efficient. A clean and un-cluttered database will directly translate into a more efficient instance to manage.

Two Tiers of Storage

In the Oracle Service Cloud, there are two tiers for storage for incident data: the database tier and the incident archive tier. Provided with every Service Cloud instance, the Oracle Service Cloud includes an incident archiving service. Incident archiving functionality is meant for moving closed incidents out of the database once the incidents are no longer needed in the database. Removing old incidents allows you to keep your database lean. If you choose not to use the archiving service, your alternative is to delete closed incidents after some period of time (configured by you). Be sure you consider any regulations that may require you to retain closed incidents for an extended period of time.

The database is architected for performance. Oracle recommends using the database for timely and interactive work going on between your customers and your agents. The Oracle Service Cloud includes analytics capabilities, but analytics can only report on data within the database. As the database grows, analytics performance can suffer. When there are fewer records in the database, analytics reports may return results quickly. However, as database tables grow with stale data, these same analytic reports may take longer to complete. Also, the database tier is the most expensive. Cost-conscious customers implement a data management strategy that aligns with their cost-benefit analysis.

By comparison, incident archive storage is less expensive and exclusively reserved for incident data that has been archived. When closed incidents are archived, this data is removed from the production database. The benefit is this data is now out of the way of your active incidents so they will no longer impact database performance. Archived incidents are still accessible from the agent console using the archive console plug-in. The archive console provides the ability to search across multiple criteria, and download selected incidents. Analytics capabilities are not available for incidents that have been archived.

The best practice for utilizing these two storage tiers is to let the database represent short term storage for open and active incidents, and use incident archive for long term storage of closed incidents.

Consider a Business Intelligence System

As incidents move through lifecycle stages, most customers appreciate the ability to analyze and report on how the incidents are being managed. While Oracle Service Cloud analytics has the ability to meet low volume organizational needs, Oracle recommends that high volume customers implement a business intelligence (BI) system. High volume Service Cloud customers often use a BI system to meet their complex business requirements such as performing complex analytics, consolidating data sets from multiple sources, and analyzing large data volumes across historical time periods.

The analytics capabilities built into the Service Cloud are often sufficient for low-volume customers, and are intended primarily for the creation, activity, and closure stages. Oracle recommends to high volume customers that once an incident is closed, your goal should be to export the incident swiftly into a business intelligence solution for additional analysis. Once the incident has been exported, the customer can archive that incident and eventually remove the incident from the Service Cloud through permanent deletion.

Managing the Lifecycle of Incident Data

Incidents typically follow a lifecycle consisting of the following stages: creation, activity, closure, archival, and destruction. Throughout this lifecycle, customers should consider what their business requirements are and design a data management strategy that aligns with their strategy. In Oracle's experience, high volume Service Cloud customers are most successful when they proactively develop and implement a data management strategy that foresees their current needs as well as their projected growth. As long as you are utilizing the two tiers of Oracle Service Cloud storage properly, you are already beginning to understand and manage your incident data lifecycle. Recognizing this lifecycle and developing a strategy for handling incidents at each stage of the lifecycle will contribute to the best possible experience for your customers, agents, and other Service Cloud users. The Service Cloud includes the ability for customers to configure their data management policies to align with their data management strategy.

Ultimately, the Oracle Service Cloud should be used for transactional activity with your customers and collaboration between your customers and agents. It should not be used for warehousing your data. For this reason, archiving or deleting no more than 30 days after incident closure is strongly recommended for high volume customers. The Oracle Service Cloud can be extended to display historical data from external data sources if longer incident history from cold storage is needed for particular agent use cases.

Controlling Database Growth

The most effective method you can employ to control database growth is to follow the guidelines above regarding proper use of database and incident archive storage through incident lifecycle management. In Oracle's experience, the primary cause of excessive database growth is the creation of new incidents coupled with the retention of old, closed incidents within the production database. In addition, some system metadata tables grow as a result of data in the incident tables, whether the incidents are open or closed. These metadata tables contain data like transactions, phrases, and data to support analytics. You can effectively manage the data in these metadata tables through the ARCHIVE_INCIDENTS configuration setting. When incidents are archived, data in these system tables is automatically removed from the database.

Configuration Setting	Default	Recommended	Description
ARCHIVE_INCIDENTS	365 days	30 days or less	The default of 365 ensures eventual archiving of incidents, and this setting may be acceptable for low-volume customers, but archiving at fewer days is strongly recommended.

Archiving closed incidents as soon as possible is strongly recommended for ensuring your Service Cloud database remains lean. Although the default to archive incidents is 365 days after closure, Oracle recommends this setting be updated to reflect your data management strategy. High volume Service Cloud customers who implement a BI solution should consider setting this to 30 day, and adjust from there. Just updating the ARCHIVE_INCIDENTS setting from the default (365 days) to 180 days has the expected impact of halving the database size. As shown below, the further you reduce the archive setting, the sooner closed incidents will be archived and the smaller your database remains.

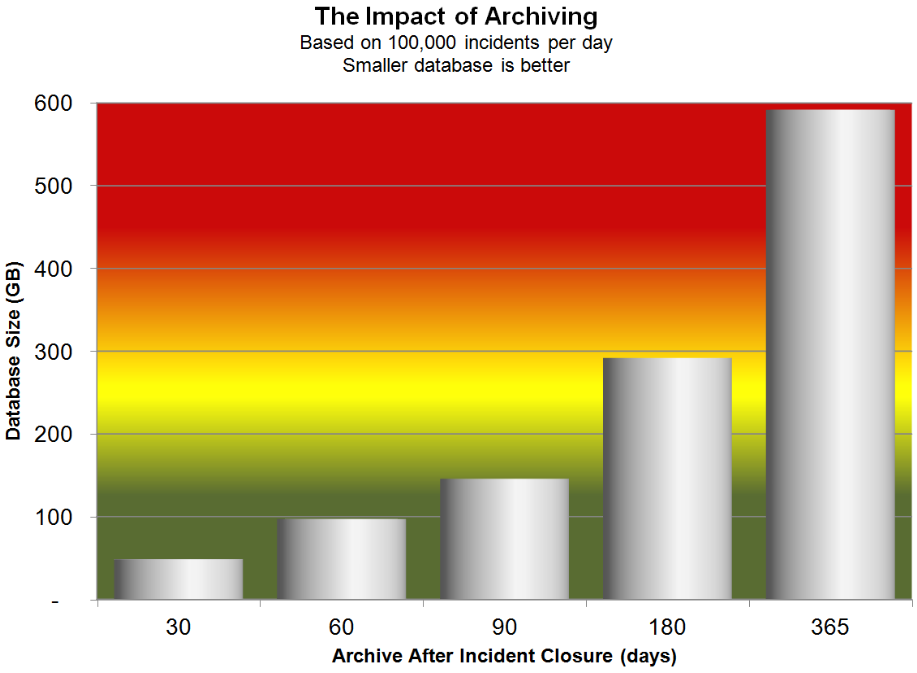


Figure 1: An example showing the impact of archiving on database size.

Contact Management

In addition to properly managing incident data, it is good practice to manage contacts, too. If possible, import contacts in an on-demand fashion. If you have a master repository of customer data, then load it into the Oracle Service Cloud when needed. Instead of pre-loading contact data into the Service Cloud, load it from your master repository at the time a customer makes contact through available channels. This has the obvious effect of reducing the amount of contact data being managed by Oracle Service Cloud, so it eliminates unnecessary clutter in the Service Cloud of customers who may never require assistance.

What Happens If You Don’t Keep a Clean House

In Oracle’s experience, the primary cause of database bloat is the retention of historical incident data in the database. While it is true you can use the rich analytics of the Oracle Service Cloud on all that data, the down side to retaining historical data includes:

- Analytics that get slower and slower over time
- Utilities that take too long to run which can delay data needed for analytics reports
- Longer times for upgrade preparation/UAT
- Longer times required for test site data refresh

- Longer times for troubleshooting when Tech Support must clone your database
- Customers or agents being negatively impacted by performance issues

Here we see the impact that database size has on clone operations. The larger the database, the more time is required to complete the task. This will be true for any operations including test data refreshes and pre-upgrade work. It is important to note that upgrade cutover time is not impacted by database size, just pre-upgrade operations require more time.

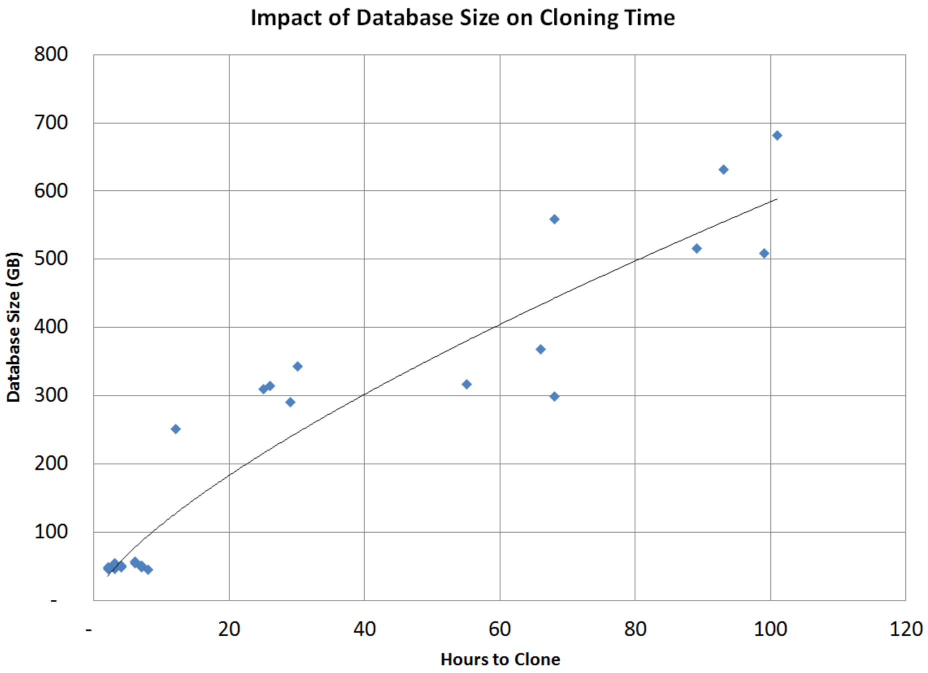


Figure 2: Impact of Database Size on Cloning Time

Recommendations

All customers should develop and implement a data management strategy. High volume customers should implement a BI system and align your incident management policies to your data management strategy. It should be clear that proper data management of your Oracle Service Cloud data is important and also achievable. In summary, Oracle recommends the following:

- Keep a clutter-free Service Cloud database
- Develop and adhere to an incident data lifecycle management plan
- Utilize the two tiers of storage for their intended purposes
- Maintain your database size through archiving
- Utilize a data warehouse tool and export incident data to it for analytics outside of the Oracle Service Cloud
- Set archiving (ARCHIVE_INCIDENTS) to 30 days or less
- Follow an assemble-on-demand model where contacts are loaded into the Oracle Service Cloud only when they are needed.

Taking these steps ensures a well-performing Service Cloud site. Your agents will perform their duties more optimally, and customers will experience efficiency when interacting with your site and your agents.

ORACLE

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

B blogs.oracle.com/cx
f facebook.com/OracleServCloud
twitter.com/OracleServCloud
oracle.com

Integrated Cloud Applications & Platform Services

White Paper Title: Best Practices for Large Deployments in the Oracle Service Cloud

July 2016
Author: Sean Montgomery
Contributing Authors: Steve Fangman

Oracle is committed to developing practices and products that help protect the environment