Database Configuration

Contents

Database Configuration and Best Practices

Make sure your databases are ready before you install or upgrade Jive.

Prerequisites

To install Jive, you'll need to set up three databases: one each for the core application, the Activity Engine, and Analytics. (While it's possible to run Jive without Analytics, any substantial installation needs to include it.) You'll find that an installation or upgrade goes more smoothly if you complete the database-related tasks first. For new installations, the database and application users must be created before installation. Follow recommended best practices for the database vendor and version selected. See the database-specific best practices below for information about how to configure your database to work with Jive.

Database Administration

We highly recommend having a skilled database administrator in charge of your Jive databases.

Locating Your Databases on Separate Servers

For large installations, it is recommended that the core application database, the Activity Engine database, and the Analytics database be hosted on separate database engines. At a minimum, disk and memory resources should be isolated for each service.

Why Separate the Core Application and Analytics Databases?

The following points explain why it's especially important for your Analytics database to be separate from the application database. Ideally, they should be stored on two separate servers. While we technically support storing the Analytics and application databases on one server, here's why it's important to ensure isolation:

- Analytics has a very different access pattern from the application database: many streaming writes, occasional reads, and busy activity once a day when the ETL runs.
- The storage requirements for Analytics are very different from the application database requirements. The physical size of the Analytics databases will grow much larger than the application database on busy systems. The IO profiles (the application database is a typical OLTP load, the Analytics database is OLAP) and the usage profiles (potentially large result sets, aggregates and ad-hoc queries are expected on the Analytics database) also support isolation of the application and analytics resources.
- The application database is more critical than the Analytics database. Therefore, the analytics database should not be in a position to demand resources that the core database may need.
- The Analytics database is intended to be accessed by users running ad-hoc queries. Because the application cannot predict which queries users may run, it is possible that a bad query could execute and thus consume all of the server's capacity.

Storing Attachments

You should always configure attachments for storage using a file system provider rather than the database. Storing attachments in the database is possible, but is known to cause severe performance issues when the document store gets large and contends with the database cache. For more information about how to set up your file system storage, see Required External Components and Configuring a Binary Storage Provider.

Database Collation

The database or database instance must be configured for UTF8 collation. See the instructions for your individual database platform for instructions.

Supported Database Engines

Jive supports recent versions of several databases.

The following database platforms are supported. For information about setting them up to deploy Jive, see all the topics under Database Configuration and Best Practices.

Jive Core and Activity Engine Databases

- Oracle 11gR2, 12c
- PostgreSQL 9.0, 9.1, 9.2, 9.3, 9.4 (9.2 or higher recommended)
- MySQL 5.1, 5.5, 5.6
- SQL Server 2008R2, 2012, 2014

Jive Analytics (Community Manager Reports) Database

- Oracle 11gR2, 12c
- PostgreSQL 9.0, 9.1, 9.2, 9.3, 9.4 (9.2 or higher recommended)

Setting Up New Databases

When setting up new databases, make sure you ensure sufficient isolation, and set an appropriate number of connections.

During the installation process, you'll need to create up to three databases: the core application database, the Activity Engine database, and (optionally) the Analytics database. These systems should be installed on separate, isolated resources, as described in detail here. Although you may be tempted to ignore this recommendation if you have a robust, high-capacity system, you should still seriously consider IO and memory isolation. It's also very important to maintain isolation of security roles.

Note: Make sure you look at the system-specific best practices for your database before installing.

The maximum number of incoming connections must be set to accommodate the sum of max connections across all web servers, plus a small administrative overhead.

For example, if 5 application nodes are set to from 50 to 150 max connections, the database should be able to allow as many as (5*150)=750 connections from the application DB, +n for administrative connections should all connections be in use. A typical overage is +5-10 connections.

Analytics databases only require connections for the application server listeners and administrative users. Typically, the maximum connections number is much lower than the application database requires. The Activity Engine database default is a maximum 50 connections. This typically does not need to be increased.

Upgrading a Database

If you plan to leave the database on the same system, no special configuration is required. If you're moving to a new database engine or system, migrate the system first, then upgrade.

If you're upgrading to a version whose database schema is different from the previous version, the application will automatically detect the difference after you've upgraded. When you next start the application and navigate to the Admin Console, you'll be prompted to start the database upgrade. This is a necessary step before you can start using the application after upgrade. You don't need to run any database-related upgrade scripts unless you are moving to a different database engine.

To upgrade the database:

1. Shut down all the services.
2. If you're using the same database engine (for example, moving SQL Server to SQL Server), back up and restore the database to the new instance.
3. If you're using a different database engine after the upgrade (for example, moving Oracle to PostgreSQL), you'll need to use scripts and ETL tools to build the database and transfer the data. Refer to Persistent Properties for Database Recovery or Migration to learn about the properties you need to pay attention to when migrating a database.
4. Start new web servers and point them to the new database engine.
5. After you restart Jive, the upgrade process will begin, and the database schema will be upgraded.

Postgres Database Best Practices

For best results, configure and maintain your Postgres database according to the vendor's requirements and use the Jive best practices.

- Configure the PostgreSQL `ph_hba.conf` to allow md5 communication between the application servers and the databases. This setting should be applied to all instances.
- Create users for each of the databases. Typically, the user names will be identified as the DB names.
- Create the databases using the OWNER option, using the applicable role created previously.
- For performance and troubleshooting, set the `logging_collector` to on, and ensure that postgres logs are time-stamped. Use caution in busy environments with log settings, because excessive logging can generate critical load on the server.

- The PostgreSQL `postgresql.conf` file is, by default, set to run using extremely limited resources. Settings where default values can impact performance include, but are not limited to, the following:
  - shared_buffers
  - checkpoint_segments
  - checkpoint_completion_target
  - work_mem
- Configure monitoring on the system to assess the following metrics, at a bare minimum
  - DB Size
  - Available Memory
  - Connections
  - Read/Write Activity
  - Transaction rates

Oracle Database Best Practices

For best results, configure and maintain your Oracle database according to the vendor's requirements and use the Jive best practices.

- If you're setting up new systems, the Core and Activity Engine databases can be set up as typical OLTP databases. Set up the Analytics database using the OLAP template.
- Oracle JDBC Drivers must be installed on the application services before the Jive installation. For more information, see Oracle Drivers.
- To ensure good performance, set the following jvm property on each of the web application nodes:

  ```
  -Doracle.jdbc.maxCachedBufferSize=12
  ```
- Ensure that the target database allows the Jive application's setup tool access to create tables. After you install the Jive application, you'll finish installation (or upgrade) by using Jive's admin tool to connect to the target database. The admin tool will create or upgrade the tables it needs. So, you need to ensure that permission to create tables is granted, at least until the upgrade or installation is completed and the application has been restarted.

  For example, if you'll be using an Oracle database for a new installation, you'll be creating a user representing a schema. That schema will be empty until the Jive application's admin tool creates tables in it. Here's an example of a script for creating such a user:

  1. `CREATE USER coredb IDENTIFIED BY changeme DEFAULT TABLESPACE jivedata;`
  2. `GRANT CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SEQUENCE, CREATE PROCEDURE, CREATE SYNONYM, CREATE TYPE to coredb;`
  3. `ALTER USER coredb QUOTA UNLIMITED ON jivedata;`

  The username can be any legal Oracle identifier -- "coredb" is just an example. The application does not require a dedicated tablespace, but many DBAs have that practice. The tablespace must be created separately. For more information about the CREATE USER statement, see the Oracle documentation.

Caution:

The database user must not be granted view access to any schema other than the one it owns. During upgrades, the application will read metadata about its tables defined in the database. Since the application does not know what the default schema is, it may inadvertently retrieve information about its tables defined in other schemas. If you have set up application instances using other schemas in the same database, this can result in erroneous information being passed to the application.

If you absolutely must use a database user with access to other schemas, ensure that there is only one set of application tables defined across all schemas to prevent this object name conflict.

.

Oracle Database Drivers

In addition to the following recommendations, be sure and follow vendor documentation for driver installation best practices.

Table 1:

Component	**Required Driver Version**	**Notes**

Web application	v11.2.0.4 OCI (ojdbc6.jar)	Type the following from a command line as the jive user:
		`jive set webapp.custom_ld_library_path_additions path/to/driver`
		`jive set webapp.custom_classpath_additions path/to/your_JDBC_Driver.jar`
		where *path/to/driver* is the path to the Oracle JDBC driver and *your_JDBC_Driver.jar* is the name of the JAR file.

Activity Engine	v11.2.0.4 (THIN) (ojdbc6.jar)	Type the following from a command line as the jive user:
		`jive set eae.custom_ld_library_path_additions path/to/driver`
		`jive set eae.custom_classpath_additions path/to/your_JDBC_Driver.jar`
		where *path/to/driver* is the path to the Oracle JDBC driver and *your_JDBC_Driver.jar* is the name of the JAR file.

Additional Configuration for Oracle 11gR2 and 12

Customers using Oracle 11gR2 and 12 need to add a `fetchSizeSupported` entry in the `database` element of the `jive_startup.xml` file as shown: <jive> <setup>true</setup> <temp/> <locale> <characterEncoding>UTF-8</characterEncoding> </locale> <database> <defaultProvider> <password encrypted="true"></password> <connectionTimeout>60</connectionTimeout> <maxConnections>50</maxConnections> <username>username</username> <driver>oracle.jdbc.driver.OracleDriver</driver> <minConnections>25</minConnections> <serverURL>jdbc:oracle:oci:@host:1521/db</serverURL> </defaultProvider> <fetchSizeSupported>false</fetchSizeSupported> </database> <connectionProvider> <className>com.jivesoftware.base.database.DefaultConnectionProvider</className> </connectionProvider> </jive>

Required Libraries for RHEL7

Note that distributions based on RHEL7 do not always install libaio.so by default. This library is required to use the Oracle driver. On app servers running Red Hat 7 or CentOS 7 that use Oracle as a database, make sure to run the following command:

```
# apt-get update && apt-get install libaio1</codeblock>
```

Microsoft SQL Server Database Best Practices

For best results, configure and maintain your Microsoft SQL database according to the vendor's requirements and use the Jive best practices.

Installation

- SQL Server installations may be standalone instances or clustered. Mirrors with automatic-failover are not supported, but you can use mirrors for redundancy or DR purposes.
- The core application and the Activity Engine will run using the jTDS MSSQL Driver version 1.3.x or higher. The Analytics database is not supported on MSSQL.
- Set your classpath to include the jTDS Driver by typing the following from a command line as the jive user:

  ```
  jive set webapp.custom_classpath_additions path/JDBC_Driver.jar
  ```

  where path/to/driver is the path to the jTDS JDBC driver and JDBC_Driver.jar is the name of the JAR file.

Note:

If you have implemented SSL to secure traffic between the web application nodes and the database, an issue with the jTDS driver will prevent the Jive application from starting. To work around this problem, run the following commands on each web application node:

```
jive set webapp.custom_jvm_args " -Djsse.enableCBCProtection=false"
jive set eae.custom_jvm_args " -Djsse.enableCBCProtection=false"
```

Case Sensitivity

SQL Server instance default collation is case-insensitive, but accent-sensitive. If database collation is not specified, the instance collation will be used. **Latin1_General_CI_AS** is recommended for typical installations. In special cases where the Jive Platform is configured to allow case-sensitive user names (for example, where jsmith and JSmith would be different logins), **Latin1_General_CS_AI** is required so that case sensitivity may be utilized in indexing and string searches.

You'll find more information on adjusting SQL Server settings in the SQL Server documentation: Setting and Changing the Server Collation Setting and Changing the Database Collation.

MySQL Database Best Practices

For best results, configure and maintain your MySQL database according to the vendor's requirements and use the Jive best practices.

UTF-8 Requirement

Jive requires the UTF-8 character set. Use the following steps to ensure it's correctly set up:

1. Use the following command to set the database default character set to UTF-8:

   ```
   ALTER DATABASE jivecoredb DEFAULT CHARACTER SET utf8;
   ```
2. After installation, add the following to the `<database>` section of the `jive_startup.xml` file:

   ```
   <mysql><useUnicode>true</useUnicode></mysql>
   ```

   Ensure that the Character Set setting in the Admin Console under **System** > **Management** > **Locale** is set to UTF-8.

Installing the Driver

- The core application and the Activity Engine will run using the MySQL JDBC Connector/J Driver. The Analytics database is not supported by MySQL.
- Set your classpath to include the MySQL JDBC Connector/J Driver by typing the following from a command line as the jive user:

  ```
  jive set webapp.custom_classpath_additions path/JDBC_Driver.jar
  ```

  where path/to/driver is the path to the MySQL JDBC driver and JDBC_Driver.jar is the name of the JAR file.

Other Best Practices with MySQL

Default configuration can vary significantly by version, so it is important to consider the following:

- The default storage engine and all jive tables should use the InnoDB storage engine. Earlier versions of MySql used BDB\MyISAM as the default storage engine.
- Ensure that the InnoDB Buffer is adequately sized. For dedicated installations with InnoDB storage engines, the buffer is typically set to 80% of the total server available memory.

- Storing attachments in your Jive database is not recommended. If you must do this, ensure that the maximum attachment size is allowed in the MySql configuration. See Packet Too Large in the MySQL 5.1 Reference Manual for more information.
- Case Sensitivity - To avoid upgrade problems, use lower-case table names in MySQL and set the `lower_case_table_namessystem` variable to 1 in the MySQL configuration. See Server System Variables in the MySQL 5.0 Reference Manual for more information.

Quick Database Setup for Evaluations

You can use the quick database setup procedure for evaluations and other "light" installations.

You can use the following steps to install PostgreSQL and create the databases you need to evaluate Jive. You'll need to create Core, Activity Engine, and Analytics databases. Use the full database instructions to install your production databases.

1. Browse Yum.postgres.org to find a postgreSQL 9.0 or higher package to use for your Jive evaluation.
2. Download the appropriate rpm for your operating system to configure the proper yum repository:

   For example, type `wget http://yum.postgresql.org/9.1/redhat/rhel-5-x86_64/pgdg-centos91-9.1-4.noarch.rpm`
3. Install the yum repository through RPM:

   For example, type `rpm -ivh pgdg-centos91-9.1-4.noarch.rpm`
4. Install PostgreSQL

   For example, type `yum -y install postgresql91-server`
5. Initialize the database:

   For example, type `/etc/init.d/postgresql-9.1 initdb`
6. Change the pg_hba.conf. For example, type:

   ```
   cat <<EOF > /var/lib/pgsql/9.1/data/pg_hba.conf
   local all all ident
   host all all <application server IP address>/32 md5
   EOF
   ```
7. Start the server:

   For example, type `/etc/init.d/postgresql-9.1 start`
8. Set up the databases and users. For example, type:

   ```
   cat <<EOF | su - postgres -c psql
   create user core with password 'core';
   create database core owner core encoding 'UTF-8';
   create user eae with password 'eae';
   create database eae owner eae encoding 'UTF-8';
   create user analytics with password 'analytics';
   create database analytics owner analytics encoding 'UTF-8';
   EOF
   ```