System Requirements

In order to install and run the database within this directory, you need to install Oracle Database 11gR2 XE. This requires a system with the following characteristics:

Linux 64-bit

11gR2 is officially supported on the following Linux operating systems:

- Oracle Enterprise Linux 4 Update 7 / OEL Linux 5 Update 2
- Red Hat Enterprise Linux 4 Update 7 / RHEL Linux 5 Update 2
- SUSE Linux Enterprise Server 10 SP2
- SUSE Linux Enterprise Server 11

> Note: it's possible to install Oracle Database Server 11gR2 on other 64-bit Linux OSes. There are install guides available online for CentOS, Ubuntu, etc. I was able to install and run 11gR2 XE on CentOS 6.4.

The server requires the following (from http://docs.oracle.com/cd/E17781_01/install.112/e18802/toc.htm#XEINL106)

- At minimum, 256MB RAM (at least 512 recommended)
- 1.5GB minimum disk space
- The following packages:
  - `glibc` at v2.3.4–2.41 or higher
  - `make` at v3.80 or higher
  - `binutils` at v2.16.91.0.5 or higher
  - `gcc` at v4.1.2 or higher
  - `libaio` at v0.3.104 or higher
- 2GB minimum swap space (or twice RAM, whichever is greater)
- The following kernel parameters must be set:
  - `semmsl` = 250
  - `semmns` = 32000
  - `semopm` = 100
  - `semmni` = 128
  - `shmmax` = 4294967295
  - `shmmni` = 4096
  - `shmall` = 2097152
  - `file-max` = 6815744
  - `VERSION` = 2.4.21
  - `ip_local_port_range` = 9000–65500
- You must have root permissions in order to install Oracle.

Windows (32-bit or 64-bit)

The following system requirements must be met (from http://docs.oracle.com/cd/E17781_01/install.112/e18803/toc.htm#XEINW105):

For Windows 32-bit:

- System architecture: Intel (x86), AMD64, Intel EM64T
- OS:
  - Windows Server 2008 (Standard, Enterprise, Datacenter, Foundation, or Web editions)
  - Windows 7 (Professional, Enterprise, or Ultimate edition)
  - Windows 8 (Pro or Enterprise)
- 256MB minimum RAM (512MB suggested) – this is for Oracle XE's requirements – your OS may have a different minimum requirement
- 1.5 GB Disk space

For Windows 64-bit:

- System architecture: AMD64, Intel EM64T
- OS:
  - Windows Server 2008 x64 - Standard, Enterprise, Datacenter, Web, or Foundation Editions
  - Windows Server 2008 R2 x64 - Standard, Enterprise, Datacenter, Web, or Foundation Editions.
  - Windows Server 2012 x64 - Standard, Datacenter, Essentials, or Foundation Editions
  - Windows Server 2012 R2 x64 - Standard, Datacenter, Essentials, or Foundation Editions
  - Windows 7 x64 - Professional, Enterprise, or Ultimate Editions.
  - Windows 8 - Pro or Enterprise Editions
  - Windows 8.1 - Pro or Enterprise Editions
- 256MB minimum RAM (512MB suggested) – this is for Oracle XE's requirements – your OS may have a different minimum requirement
- 1.5 GB Disk space
- Windows Installer 2.0 or later

If you intend on using the Windows Firewall, ensure the following ports are open:

- 1521 - Listener
- 2030
- 8080 - Oracle HTTP Transaction Server

In order to install, you must be a member of the Administrators group.

Downloading Oracle Database 11gR2 XE

You may obtain the download for your operating system from the following link:

- http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index-083047.html

You will need to agree to the license agreement and create a free Oracle Technology Network account prior to downloading the installation package.

Installing Oracle 11gR2 XE

Follow the instructions for your operating system from the following guides:

- For Linux 64-bit: http://docs.oracle.com/cd/E17781_01/install.112/e18802/toc.htm#XEINL106
- For Windows 32-bit or 64-bit: http://docs.oracle.com/cd/E17781_01/install.112/e18803/toc.htm#XEINW105

Installing on CentOS 6.4

Although not officially supported, CentOS 6.4 is close enough to Red Hat Enterprise Linux and runs Oracle very well. The following instructions should work to get an instance running on your CentOS system:

1. Log in as `root`
2. Install required pre-requisites:

   ```
   yum install man bc xauth wget gcc gcc-c++ automake autoconf libtoolize make
   ```

3. Add the `dba` group:

   ```
   groupadd dba
   ```

4. Add the `oinstall` group:

   ```
   groupadd oinstall
   ```

5. Add the `oracle` user:

   ```
   useradd -g dba -G oinstall oracle
   ```

6. Assign a password to `oracle`:

   ```
   passwd oracle
   ```

7. Make sure `oracle` has rights to `sudo`:

a. Launch `visudo`

b. Find the line that looks like this (`root ALL=(ALL) ALL`) and add the following below it:

```
    oracle     ALL=(ALL)       ALL
```

8. Set up a swap file:

```
dd if=/dev/zero of=/swapfile bs=1024 count=2048k
mkswap /swapfile
swapon /swapfile
chown root:root /swapfile
chmod 0600 /swapfile
sysctl vm.swappiness=30
vi /etc/sysctl.conf
# Add or alter vm.swappiness to 30
vm.swappiness=30
:wq
vi /etc/fstab
# add the following line
/swapfile          swap            swap    defaults        0 0
# execute these commands to save and exit VI
:wq
```

9. Log in as `oracle`, and `unzip` the file you downloaded from Oracle. It should create a `Disk1` directory.

10. `cd Disk1`

11. Install the `rpm`:

```
sudo rpm -ivh oracle-xe-11.2.0-1.0.x86_64.rpm
```

12. Configure the installation – You'll be prompted for Oracle account passwords and a port number for Application express. For the latter I used `8080`. You'll also be prompted for a listener port – the default is 1521. It's better to use a nonstandard port. I used `52300`.

```
sudo /etc/init.d/oracle-xe configure
```

13. Copy the Oracle environment shell script to Oracle's home directory:

```
cp /u01/app/oracle/product/11.2.0/xe/bin/oracle_env.sh ~/oraenv.sh
```

14. Set up your environment by sourcing `oraenv.sh`:

```
. ~/oraenv.sh
```

> Note: You may need to alter your `/etc/hosts` file to associate your server's hostname with `localhost`.

About the Database

The database engine is Oracle 11gR2 (Express Edition). This should work on any Oracle edition 11gR2 or later. At the time of this writing, 12c is not available in an Express Edition.

> NOTE: This is *not* how I would suggest structuring your database in a production enterprise environment. For educational purposes, the data and API tiers are combined, but in a real environment, I would break these out into multiple tiers – the data living in one schema, heavily protected, the APIs living in another two schemas (security and app APIs), and a separate layer for views. The idea would be to segregate each layer such that it would be possible to allow access to the data via views to typical user accounts while also allowing access to the more privileged code and data using very specific and controlled accounts. How to do this is beyond the scope of this project, and varies by database platform.

Installing Tasker

Once the database has been installed (see above), follow the following steps to install Tasker into the database:

1. Log in to the database:

   ```
   sqlplus / as sysdba
   ```

2. Create a new tablespace named `TASKER`:

   ```
   create tablespace TASKER logging
   datafile '/u01/app/oracle/oradata/XE/tasker.dbf'
   size 64m autoextend on next 64m maxsize unlimited
   extent management local;
   ```

3. Create a new user called `TASKER`

   ```
   create user tasker
   identified by 'a-password'
   default tablespace tasker
   temporary tablespace temp;
   ```

4. Assign the following privileges to TASKER (from a SYSDBA):

```
grant create any context to tasker;
grant execute on dbms_session to tasker;
```

5. Copy the SQL files in this directory to your `/home/oracle` on your database server.

6. Run the following SQL files to create the various objects (as the `Tasker` user):

```
connect tasker/tasker-password;
@sequences.sql
@tables.sql
@triggers.sql
@app_settings.sql
@session_context.sql
@utils.sql
@security.sql
@user_mgmt.sql
@person_mgmt.sql
@task_mgmt.sql
```

7. Create the necessary context as the `Tasker` user:

```
create context tasker_ctx using session_context accessed globally;
```

8. Execute the `sample_data.sql` file to create some sample data as the Tasker user:

```
@sample_data.sql
exit
```

Settings

Settings are controlled in the `TASKER.SETTINGS` table.

Authentication Token Mode (`AUTH_TOKEN_MODE`)

This setting determines whether a new authentication token is regenerated upon each request.

- `REGEN_EVERY` - creates a new authentication token upon each request
- `REGEN_NEVER` - only creates authentication tokens upon user login – useful for testing

Default Password Hash Iterations (`DEFAULT_PASSWORD_HASH_ITER`)

This setting controls how many times a password hash is iterated. The default value is `16384` .

The higher this setting, the longer a password check will take, but the more secure the system will be. Higher values are suggested for production (around `100000`).

Token Hash Iterations (`TOKEN_HASH_ITER`)

This setting controls how many times the authentication token is iterated. The default is `256`.

As with password hashes, the higher this number, the more secure, but at a price: requests to the system will be much slower.

Token Hash Length (`TOKEN_HASH_LENGTH`)

This setting controls how large each portion of a client token is. The token is created in two parts, so the default value of `128` generates a token of `256` bytes. Because these bytes are then rendered as hex strings, the actual result is a `512` character long token.

Password Salt Length (`PWD_SALT_LENGTH`)

Controls how long the password salt is. Default is `48` bytes.

Password Hash Length (`PWD_HASH_LENGTH`)

Controls how long the password hash is. Default is `128` bytes.

Maximum Failed Login Attempts (`MAX_FAILED_LOGINS`)

Determines how many failed logins are acceptable before locking a user account. Default is `10` failed attempts.

Token Expiry (`TOKEN_EXPIRY`)

Determines how many days a token is valid for – the default is `7`. Tokens can always be immediately expired if the user logs out, or by expiring the token from the `SESSIONS` table.

Tables

- `PERSON` : stores personal information, like name and administrator.
- `ROLES` : lists specific roles granted to specific role names, like `SYSADMIN`
- `SESSIONS` : stores session information, including token, HMAC secret, etc.
- `TASK` : Task data
- `TASK_COMMENTS` : Comments for tasks
- `USERS` : Each user has a record here along with a salted password hash. This table also keeps track of the number of failed login attempts. `ITER` and `LEN` columns store the number of iterations and hash length when the password was last changed so that changing the `SETTINGS` table won't invalidate existing passwords.
- `USER_ROLES` : Maps users to given roles.

Roles

The following roles are recognized:

- `CAN_CHANGE_ANY_USER_PWD` - A user with this role can change any user’s password
- `CAN_MODIFY_USER` - A user with this role can modify the user data of any user
- `CAN_CREATE_USER` - A user with this role can create new users
- `CAN_MODIFY_ANY_TASK` - A user with this role can modify any task, not just their own
- `CAN_ASSIGN_ANY_TASK_TO_ANYONE` - A user with this role can assign tasks to anyone, not just direct subordinates
- `CAN_SEE_ANY_TASK` - A user with this role can see any task, not just their own or assigned
- `CAN_COMMENT_ON_ANY_TASK` - A user with this role can comment on any task, not just their own or assigned
- `CAN_REASSIGN_ANY_TASK` - A user with this role can reassign any task
- `CAN_CREATE_OWN_TASK` - A user with this role can create their own tasks
- `CAN_MODIFY_OWN_TASK` - A user with this role can change their own tasks
- `CAN_COMMENT_ON_OWN_TASK` - A user with this role can comment on their own tasks
- `CAN_COMMENT_ON_ASSIGNED_TASK` - A user with this role can comment on any tasks assigned to them
- `CAN_UPD_PROGRESS_ON_ASGND_TASK` - A user with this role can update the progress on any task assigned to them
- `CAN_UPD_STATUS_ON_ASGND_TASK` - A user with this role can update the status on any task assigned to them
- `CAN_UNLOCK_USER` - A user with this role can unlock another user (resets failed login attempts)
- `CAN_CREATE_PERSON` - A user with this role can create new people records
- `CAN_MODIFY_PERSON` - A user with this role can modify existing people records

Push Notifications

When data is changed on an exiting task or a comment is added to a task, a push notification is automatically generated. For this reason, the following permissions must be run once:

TODO: fill in.