
ORACLE®

# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

# DevOps for DBCS

## Demo Overview

<name>
Oracle Cloud Platform Specialist

May 12th 2017

ORACLE

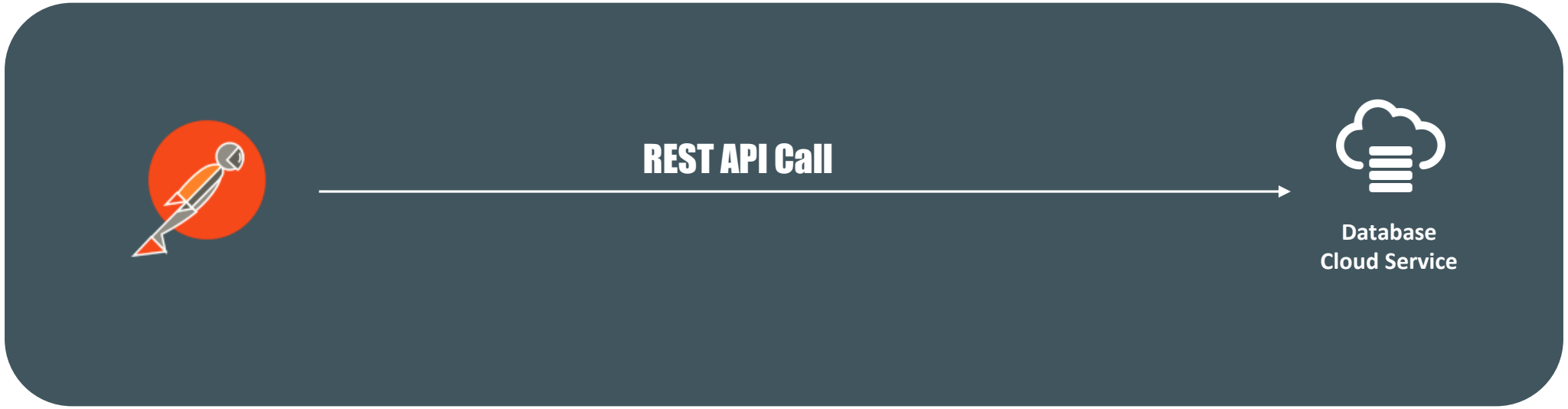
# Demo #1

## DBCS Provisioning

# What are we looking to do?

- Provision a 12c EE database
- Automate the provisioning process
  - Preferably use scripting to make it extensible
- Easily extend scripting to full lifecycle management
- Show in automated tools – ex: a build server

# Architecture Stage 1: Build a RESTful Call



# Architecture Stage 2: Parameterize with Automated Build

# How we've done it?

1. We've taken the REST API command to provision a DBCS instance
2. We've identified the parameters to this script that we want to abstract out
    1. In this case the name and credentials we don't want stored in our script
3. We load this script into an automated build as a straight shell command
4. We build out the necessary parameters which gives us an executable form when executing the build
5. We can now have a push button DB provisioning pipeline

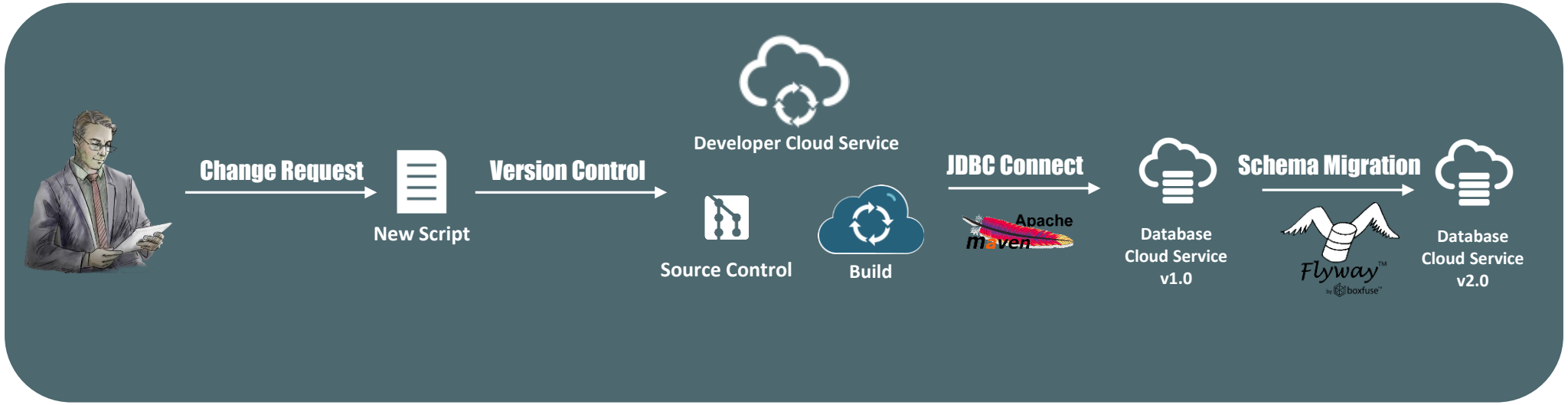
# Demo #2

## DBCS Schema Migration

# What are we looking to do?

- Build out a database schema migration pipeline
  - Intake new Schema change requests
  - Version and store new schema scripts
  - Apply new scripts and schema changes to a specific database

# Architecture: Database Schema Migration



# How we've done it?

1. New Request comes in to setup an application schema on a new DB instance
2. We have a small DB app where we place in any new scripts
3. The application leverages Maven to execute builds and a plugin, FlywayDB, to connect and migrate schemas
4. The application is version controlled in Developer Cloud Service
    1. This would allow the database source code can be branched and tagged to match application releases
    2. We can go back in time to see changes
    3. We can do issue tracking in DevCS against this application
5. An automated build is created to apply our DB script changes against a target DB
    1. This will setup schema, change recoding table and apply any scripts in the application against the database
6. We now have an automated schema migration pipeline