

# Service Brokers

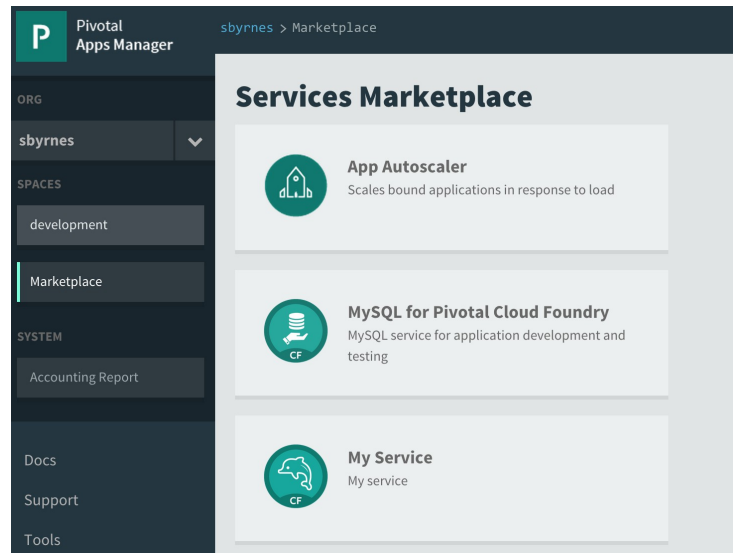
## Adding Managed Services

# Topics

- **Managed Services Overview**
- Service Broker API
- Service and Service Broker Implementation
- MySQL Service for Pivotal Cloud Foundry

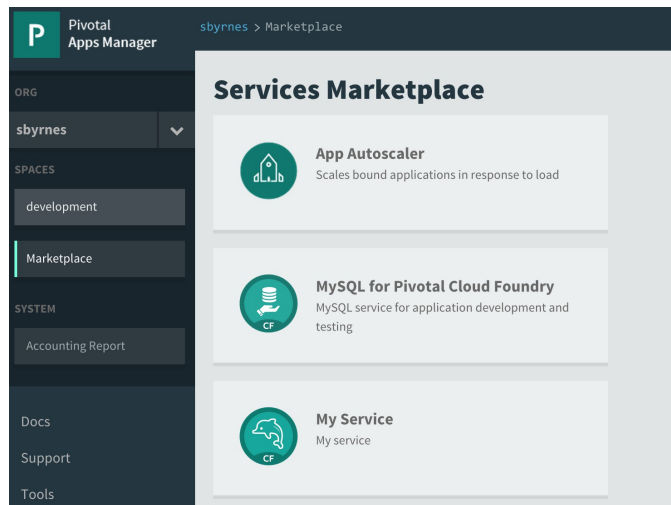
# Why Managed Services?

- A managed service is a “first class” citizen on the Cloud Foundry platform
  - Available in the marketplace
  - Has service plans
  - Can provision or reserve resources
  - Can provision credentials
  - Services instances can be created
- For operators:
  - Can add a custom service to the platform and let the developers provision them
- For developers:
  - Can provision and use services immediately and without involving operations
  - Increases agility



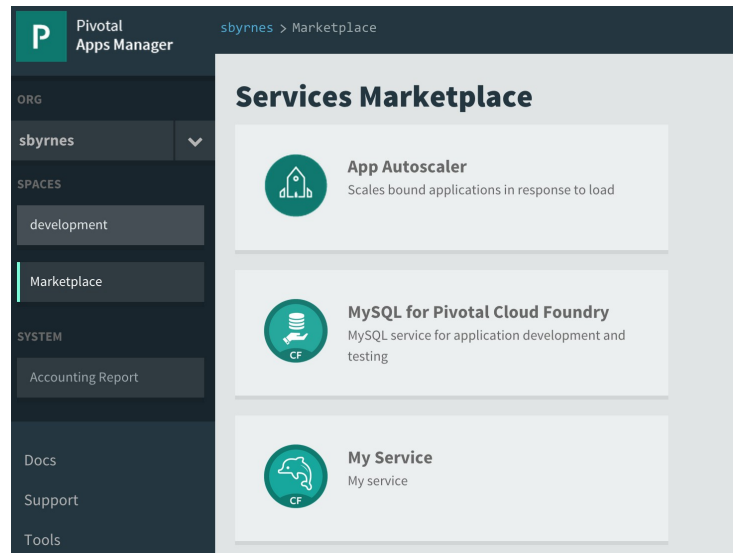
# Managed Services

- Available as plans in the marketplace
- The service *may* be a Cloud Foundry application
- The service *may* be a BOSH deployment (for example, MySQL)
- Includes a service broker, which implements a service broker API
- The service broker *may* be a Cloud Foundry application

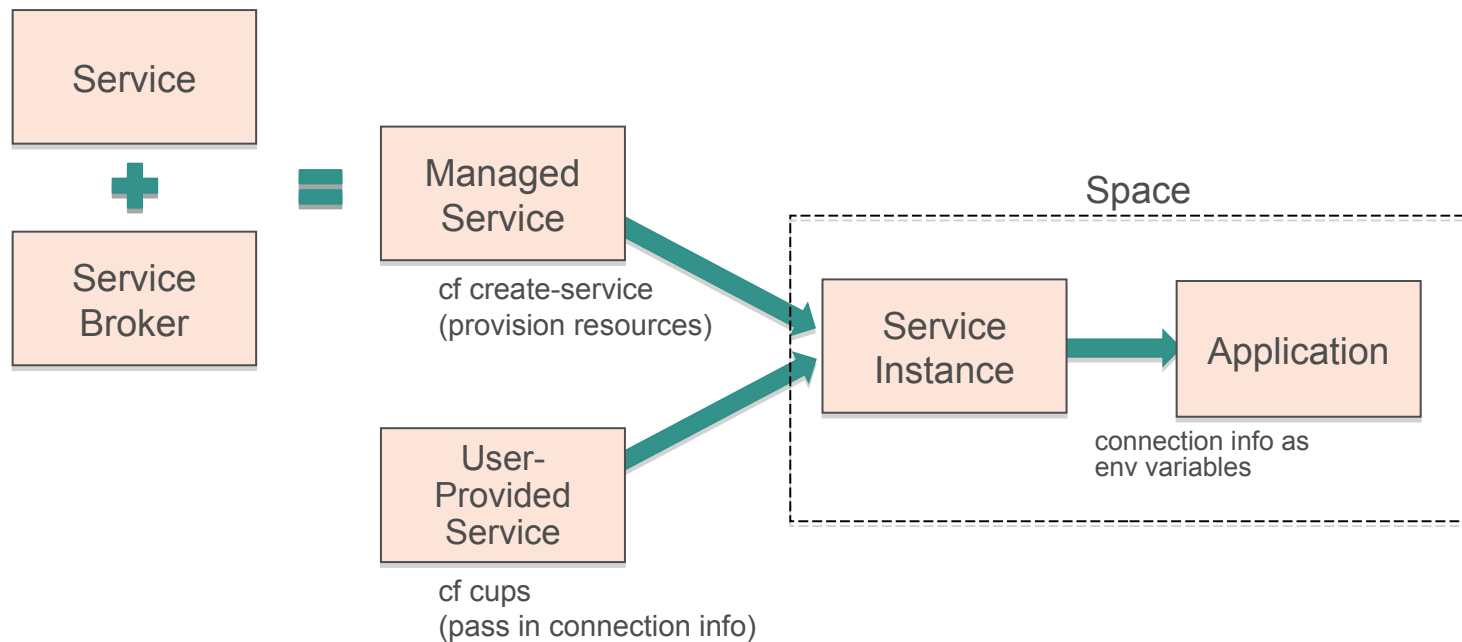


# Service Broker Overview

- Service brokers provide a standardized way to add a managed service to the Cloud Foundry platform
  - Service broker API
- Create a service broker to add a custom service to the marketplace



# The Services Big Picture

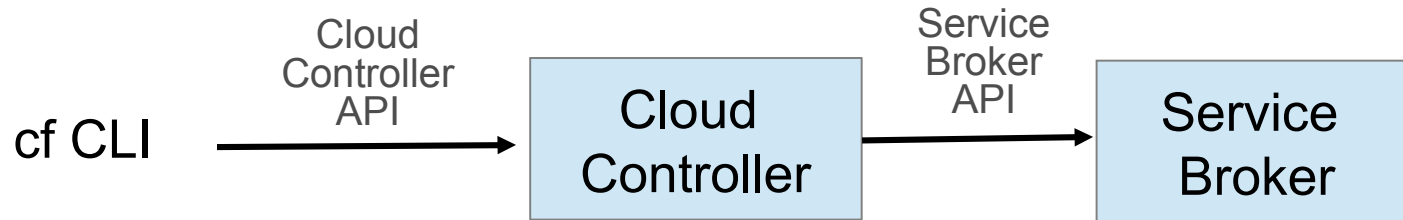


*For managed services, connection info is provisioned when binding the service instance to the application*

# Topics

- Managed Services Overview
- **Service Broker API**
- Service and Service Broker Implementation
- MySQL Service for Pivotal Cloud Foundry

# The Cloud Controller vs. Service Broker APIs



- Requests from developers or operators related to services start with the cf CLI or other client
- The cf CLI actually uses the cloud controller API
- The cloud controller calls the service broker API



# Service Broker API

The screenshot shows a web browser window with the address bar displaying `docs.cloudfoundry.org/services/api.html`. The page title is "Cloud Foundry Documentation". On the left, a sidebar menu lists various topics, with "Service Broker API" highlighted. The main content area is titled "Service Broker API v2.8" and contains a list of links under the heading "On this page:". The links include "Document Changelog", "Changes" (with sub-links "Change Policy" and "Changes Since v2.7"), "Dependencies", "API Overview", "API Version Header", "Authentication", "Catalog Management" (with sub-link "Adding a Broker to Cloud Foundry"), "Asynchronous Operations" (with sub-links "Sequence Diagram", "Blocking Operations", and "When to use Asynchronous Service Operations"), and "Operations".

docs.cloudfoundry.org/services/api.html

Cloud Foundry Documentation

Overview

**Service Broker API**

Managing Service Brokers

Access Control

Catalog Metadata

Binding Credentials

Dashboard Single Sign-On

Example Service Brokers

Application Log Streaming

Supporting Multiple Cloud Foundry Instances

## Service Broker API v2.8

On this page:

- [Document Changelog](#)
- [Changes](#)
  - [Change Policy](#)
  - [Changes Since v2.7](#)
- [Dependencies](#)
- [API Overview](#)
- [API Version Header](#)
- [Authentication](#)
- [Catalog Management](#)
  - [Adding a Broker to Cloud Foundry](#)
- [Asynchronous Operations](#)
  - [Sequence Diagram](#)
  - [Blocking Operations](#)
  - [When to use Asynchronous Service Operations](#)

# Service Broker API: /catalog

Request

Route

```
GET /v2/catalog
```

cURL

```
curl -H "X-Broker-API-Version: 2.7" http://username:password@broker-url/v2/catalog
```

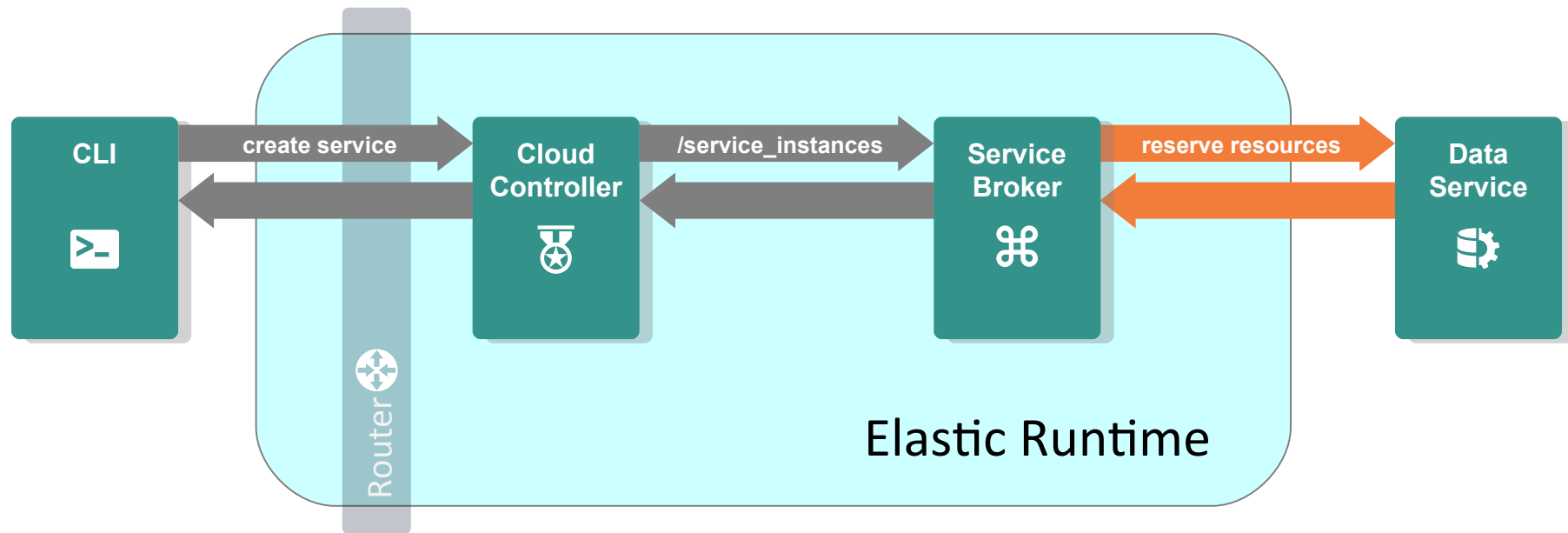
- The [/catalog](#) endpoint returns JSON with information like the service's name, description and plans
- The catalog is visible in the marketplace

# Service Broker API: /service\_instances

- **PUT /v2/service\_instances/:instance\_id**
  - Create (provision) a new service instance
  - For longer operations, use asynchronous provisioning by appending `accepts_incomplete=true`
  - The cloud controller creates and passes a unique `:instance_id` to the service broker
  - The `:instance_id` is used for future calls, such as the DELETE
- **DELETE /v2/service\_instances/:instance\_id**
  - Delete (deprovision) a service instance

# Creating a Service Instance

- The **cf create-service** command is executed, making the service instance available to the current space
  - Calls the service broker's **PUT** `/v2/service_instances/:instance_id` endpoint

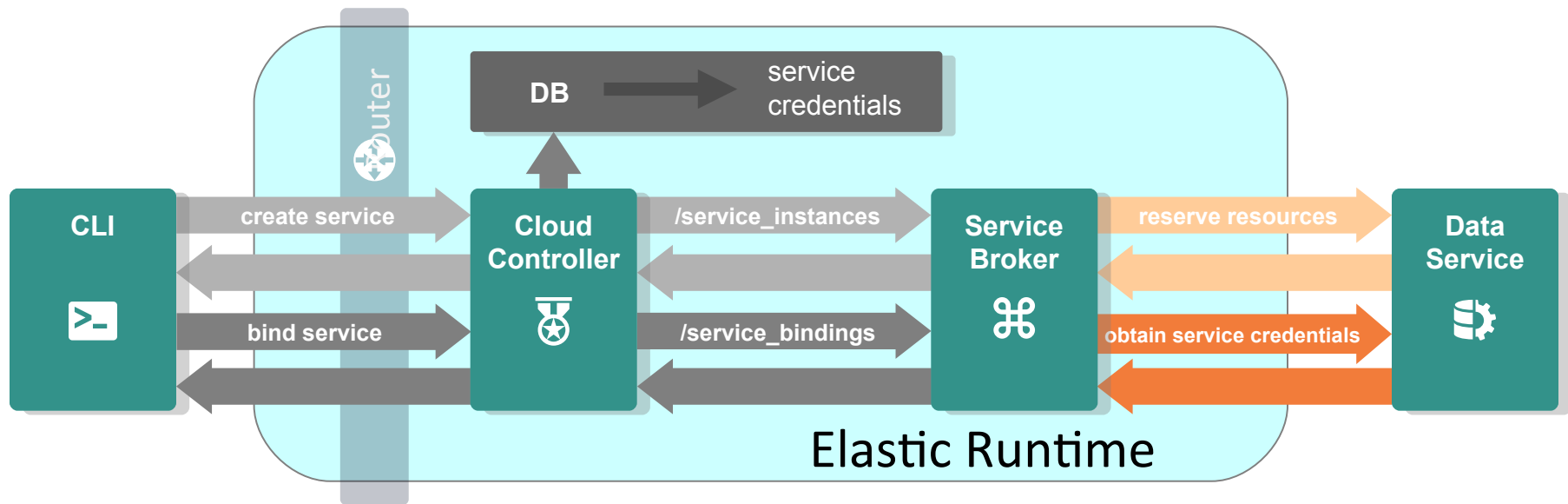


# Service Broker API: /service\_bindings

- **PUT** /v2/service\_instances/:instance\_id/service\_bindings/:id
  - Create a new binding to a service instance
  - Results in connection information passed from the service broker to the cloud controller and its database
    - The service broker *should* provision but may also reuse credentials
  - The :instance\_id is from the /service\_instances endpoint
  - The :id represents the service binding and is created and passed from the cloud controller
    - The cloud controller associates the service instance, service binding and the application
- **DELETE** /v2/service\_instances/:instance\_id/service\_bindings/:id
  - Unbind from a service instance

# Binding a Service

- The **cf bind-service** command is executed
  - Calls the service broker's endpoint- **PUT**  
`/v2/service_instances/:instance_id/service_bindings/:id`
  - Adds service credentials for the bound application to the cloud controller database
  - Those credentials are passed to application instances via environment variables



# Service Broker API- Update Service Plan

- **PATCH** /v2/service\_instances/:instance\_id
  - Update a service instance to a different plan
  - The /catalog endpoint must include **plan\_updateable: true**

# Service Broker API- Summary

- **GET /v2/catalog**
  - List services and plans available from this broker
- **PUT /v2/service\_instances/:instance\_id**
  - Create (provision) a new service instance
- **DELETE /v2/service\_instances/:instance\_id**
  - Delete (deprovision) a service instance
- **PATCH /v2/service\_instances/:instance\_id**
  - Update a service instance to a different plan
- **PUT /v2/service\_instances/:instance\_id/service\_bindings/:id**
  - Create a new binding to a service instance
- **DELETE /v2/service\_instances/:instance\_id/service\_bindings/:id**
  - Unbind from a service instance



# Topics

- Managed Services Overview
- Service Broker API
- **Service and Service Broker Implementation**
- MySQL Service for Pivotal Cloud Foundry

# Service Broker Implementation

- Service broker implementation is up to the service provider/developer
- Cloud Foundry only requires that the service provider implement the service broker API
- Deploy using **cf create-service-broker**
  - Requires admin privileges
  - The service is then available in the marketplace

# Developing a Custom Service Broker

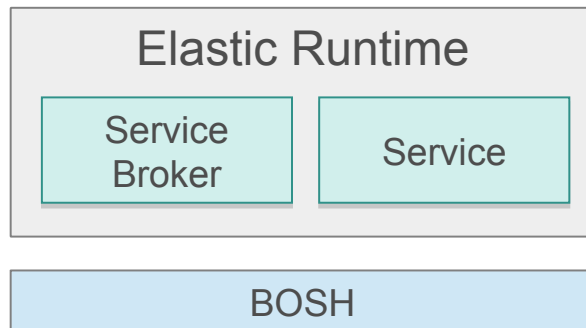
- Several GitHub projects exist
  - Use as starting points for defining a service broker
- Download generic project at
  - <https://github.com/cloudfoundry-community/spring-boot-cf-service-broker>
- Or rework Mongo DB example project
  - <https://github.com/spring-cloud-samples/cloudfoundry-service-broker>

# Service Instance Provisioning Examples

- For non-data services, provisioning could just mean getting an account on an existing system
- For the MySQL for Pivotal Cloud Foundry service, provisioning results in a database being created on the installed MySQL server
- The Cloud Foundry platform is flexible and can support many MySQL provisioning examples
  - Creating new MySQL server instances running on its own VM or in a container on a VM
  - A database with business schema already there
  - A copy of a full database, for example a QA database that is a copy of the production database

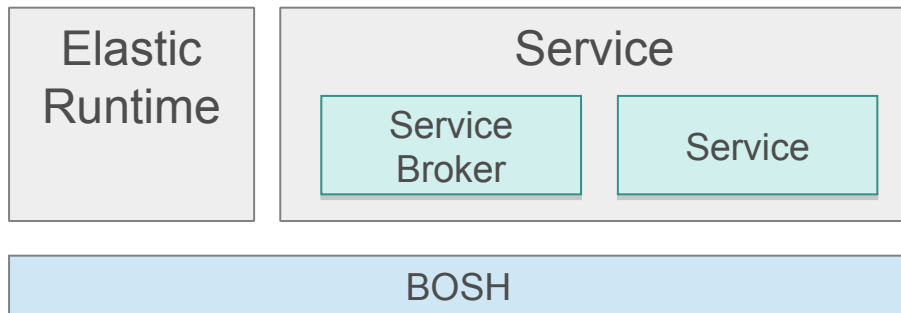
# Service and Service Broker Deployment Models (1/3)

- Service and service broker pushed as applications in an Elastic Runtime space
- Service broker endpoints can be added to an existing service



# Service and Service Broker Deployment Models (2/3)

- Entire service (and optionally service broker) packaged and deployed by BOSH alongside Cloud Foundry
- This is the MySQL for Pivotal Cloud Foundry model



# Service and Service Broker Deployment Models (3/3)

- Entire service, including service broker, deployed and maintained outside of BOSH and the Elastic Runtime



*Any combination of the deployment models will also work*

# Topics

- Managed Services Overview
- Service Broker API
- Service and Service Broker Implementation
- **MySQL Service for Pivotal Cloud Foundry**



# Example: MySQL Service

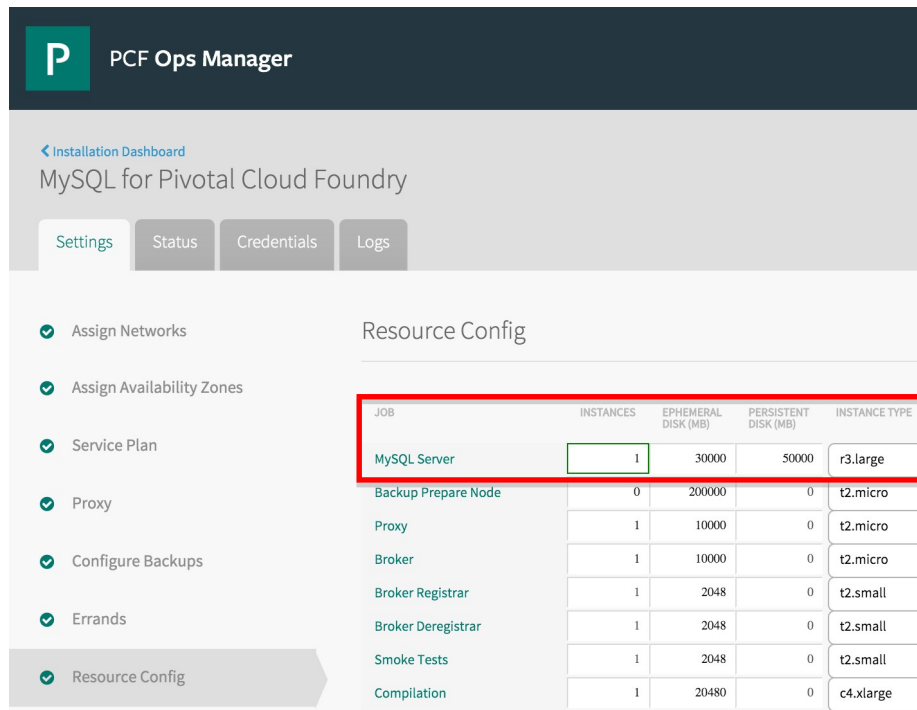
- Pivotal Web Services
  - The MySQL Server and service broker have already been installed
  - MySQL services are available in the marketplace
- For custom installations
  - Use Ops Manager to install the MySQL for Pivotal Cloud Foundry tile
    - The MySQL Server and service broker are installed using BOSH, based on settings in Ops Manager
  - MySQL services are then available in the marketplace

# MySQL for Pivotal Cloud Foundry Service Provisioning and Binding

- MySQL database provisioning (**cf create-service**)
  - The service is pre-provisioned by BOSH when the MySQL tile is installed or updated
  - When creating a service instance, a “create database” statement issued on the installed MySQL Server
- MySQL binding (**cf bind-service**)
  - MySQL user and password are created for the application
  - Credentials are passed to the application instances as environment variables
  - The user is granted access to the schema

# MySQL Server Provisioning

- Set the MySQL Server persistent disk size



PCF Ops Manager

< Installation Dashboard

MySQL for Pivotal Cloud Foundry

Settings Status Credentials Logs

Assign Networks

Assign Availability Zones

Service Plan

Proxy

Configure Backups


Errands

Resource Config

Resource Config

JOB	INSTANCES	EPHEMERAL DISK (MB)	PERSISTENT DISK (MB)	INSTANCE TYPE
MySQL Server	1	30000	50000	r3.large
Backup Prepare Node	0	200000	0	t2.micro
Proxy	1	10000	0	t2.micro
Broker	1	10000	0	t2.micro
Broker Registrar	1	2048	0	t2.small
Broker Deregistrar	1	2048	0	t2.small
Smoke Tests	1	2048	0	t2.small
Compilation	1	20480	0	c4.xlarge

# View the Current MySQL Server Disk Usage

 PCF Ops Manager admin ▾

[Installation Dashboard](#)  
MySQL for Pivotal Cloud Foundry




Settings

Status

Credentials

Logs

Jobs on Availability Zone "us-east-1a"

JOB	INDEX	IPS	CID	LOAD AVG15	CPU	MEMORY	SWAP	SYSTEM DISK	EPHEM. DISK	PERS. DISK	LOGS
MySQL Server	0	10.0.16.71	i-488416f6	0.05	0.5%	3%	0%	44%	8%	5%	
Proxy	0	10.0.16.72	i-4b8416f5	0.05	0.0%	9%	0%	44%	4%	N/A	
Broker	0	10.0.16.73	i-788517c6	0.05	0.1%	29%	0%	44%	12%	N/A	

# Service Instance Plan Size

- Each service instance reserves storage on the MySQL Server

PCF Ops Manager

[Installation Dashboard](#)

## MySQL for Pivotal Cloud Foundry

Settings Status Credentials Logs

Assign Networks

Assign Availability Zones

Service Plan

Proxy

Configure Backups

### MySQL Service Plan Configuration

Max Storage (MB) \*

100

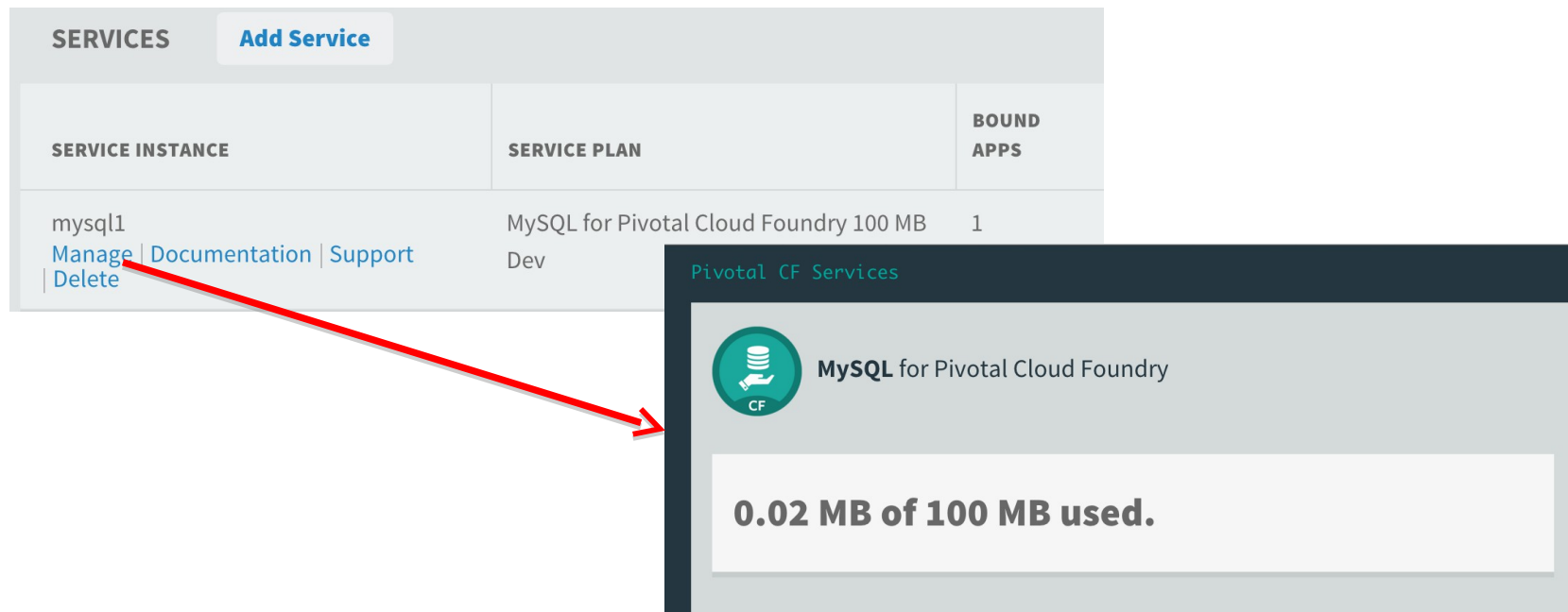
Max Concurrent Connections \*

40

This value is the maximum storage on disk permitted per database. When this quota is exceeded, write permissions are revoked until storage utilization falls below the quota. Users can discover current storage utilization on the service dashboard.

# Viewing Current Service Instance Usage

- Navigate to the service instance in a space and click the Manage link



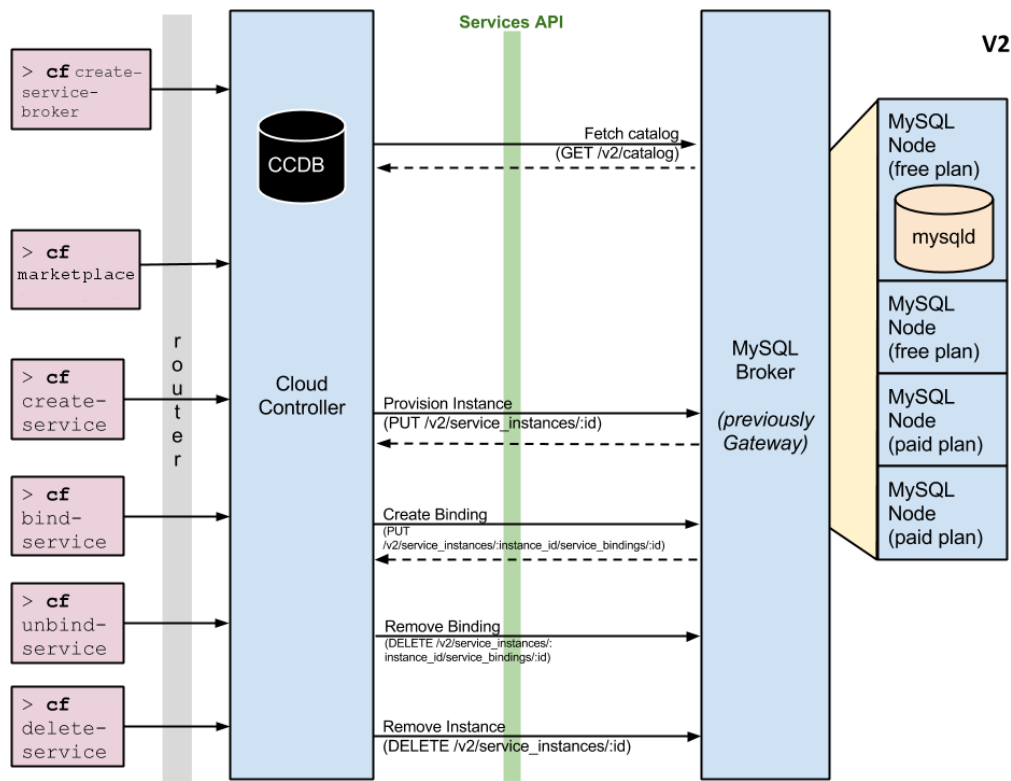
The screenshot displays the Pivotal Cloud Foundry (CF) Services interface. At the top, there's a header with 'SERVICES' and an 'Add Service' button. Below this is a table with three columns: 'SERVICE INSTANCE', 'SERVICE PLAN', and 'BOUND APPS'. The table contains one entry for 'mysql1' with the plan 'MySQL for Pivotal Cloud Foundry 100 MB Dev' and '1' bound app. Under the 'SERVICE INSTANCE' column for 'mysql1', there are links: 'Manage', 'Documentation', 'Support', and 'Delete'. A red arrow points from the 'Manage' link to a detailed view of the service instance. This detailed view, titled 'Pivotal CF Services', shows the 'MySQL for Pivotal Cloud Foundry' service with a usage summary: '0.02 MB of 100 MB used.'

SERVICE INSTANCE	SERVICE PLAN	BOUND APPS
mysql1 <a href="#">Manage</a>   <a href="#">Documentation</a>   <a href="#">Support</a>   <a href="#">Delete</a>	MySQL for Pivotal Cloud Foundry 100 MB Dev	1

**MySQL for Pivotal Cloud Foundry**

**0.02 MB of 100 MB used.**

# MySQL Service Summary



# Lab

Create, register and use a new service broker