

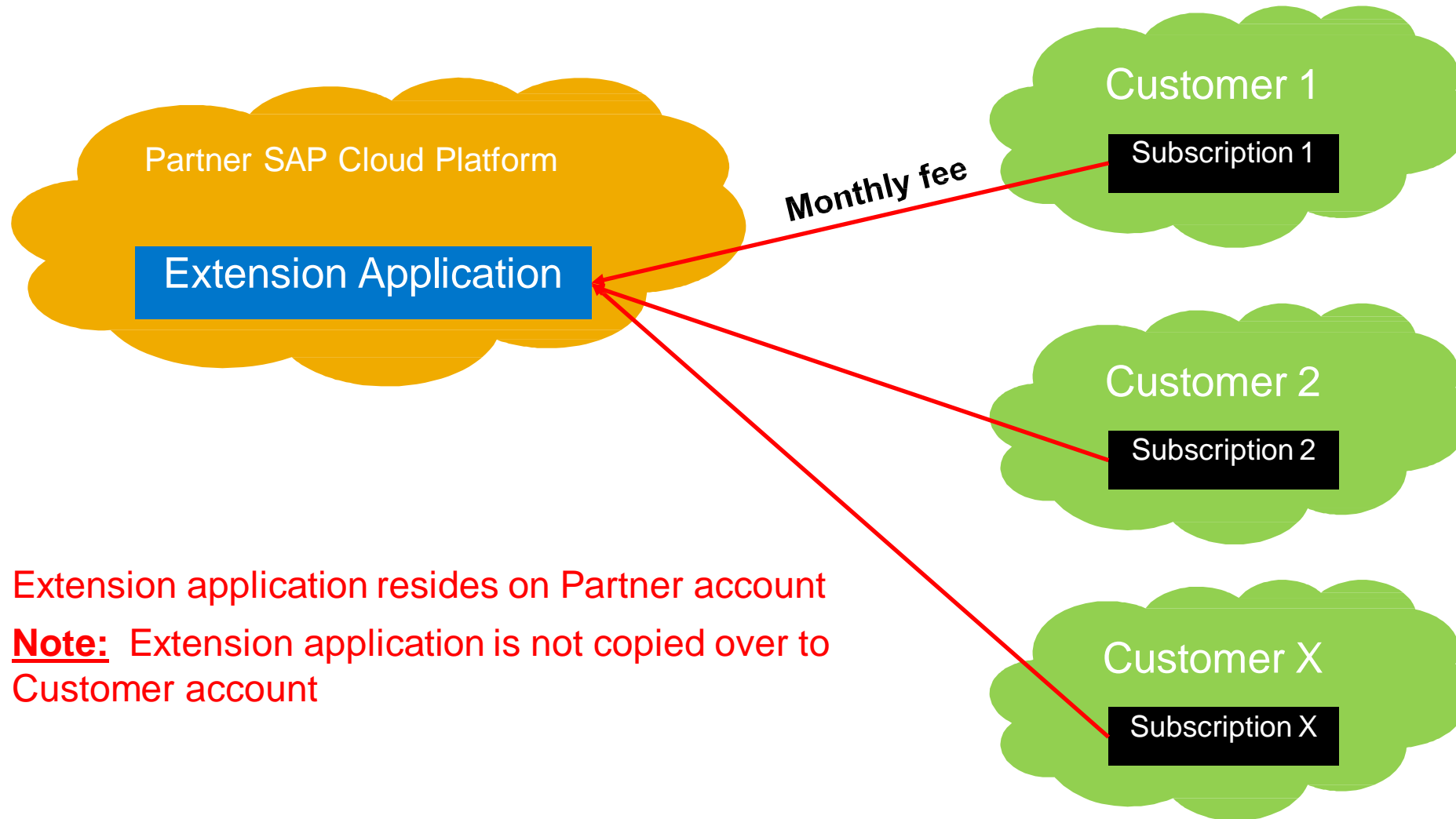


Concepts for Multitenancy

Multi-tenancy on SAP Cloud Platform

- With SAP Cloud Platform you can develop and run **multi-tenant (tenant-aware)** applications
- Applications running on a **shared compute unit** that can be **used by multiple consumers (tenants)**
- Each consumer accesses the application through a **dedicated tenant-specific URL**.

What is a Multi-Tenant application ?



Extension application resides on Partner account

Note: Extension application is not copied over to Customer account

Some basic terminology



Customer
Account

Customer = Consumer = Subscriber = Tenant = Entity paying for the subscription

Customer Account = SAP Cloud Platform account of the customer / consumer

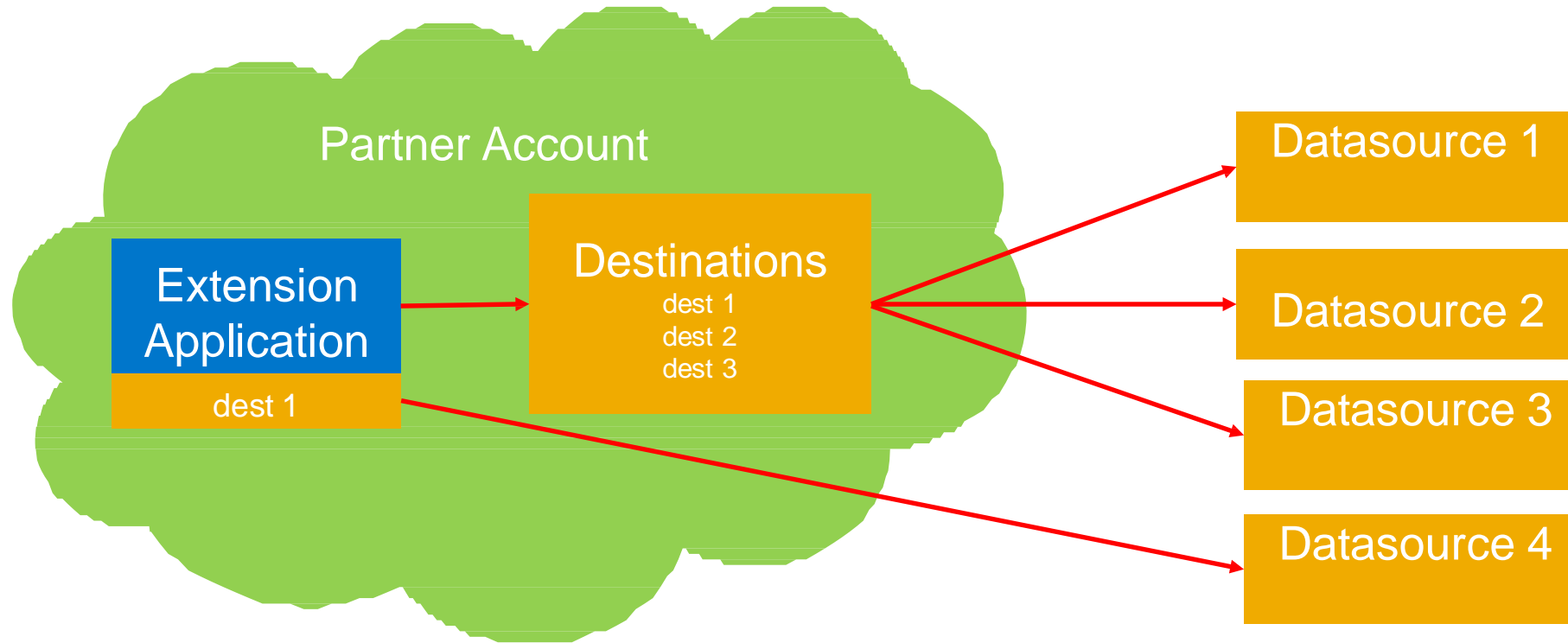


Partner
Account

Partner = Provider = Entity that created the Extension application

Partner Account = SAP Cloud Platform account of the partner / provider

Before we move on... What is a destination ?

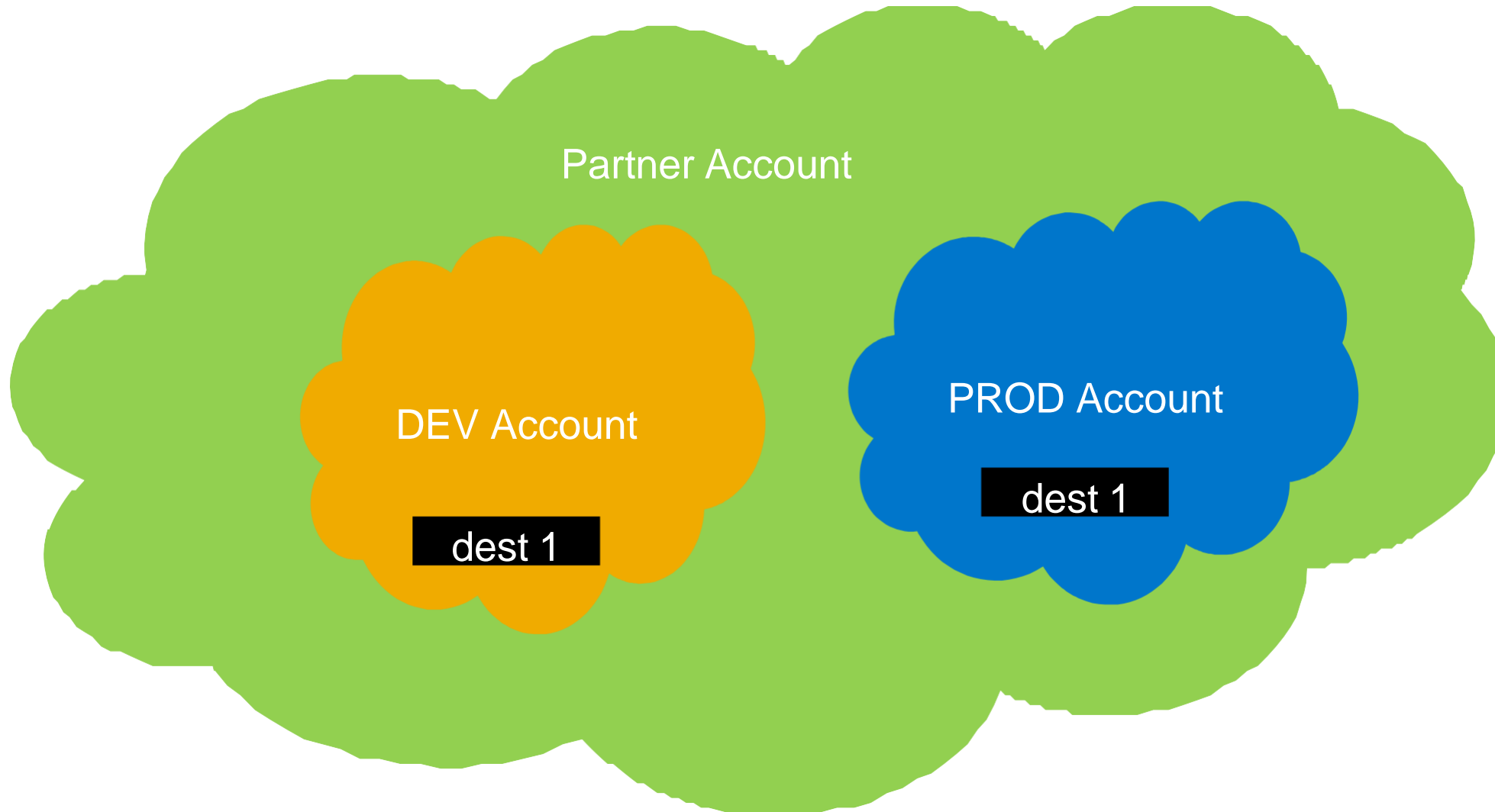


Destination = Secure connection string to a data source

Destination – Account Level (Visible for the whole account)

Destination – Application Level (Visible for the application only)

Before we move on... What is a sub account ?



Understanding SAP Cloud Platform Accounts



Each account holds:

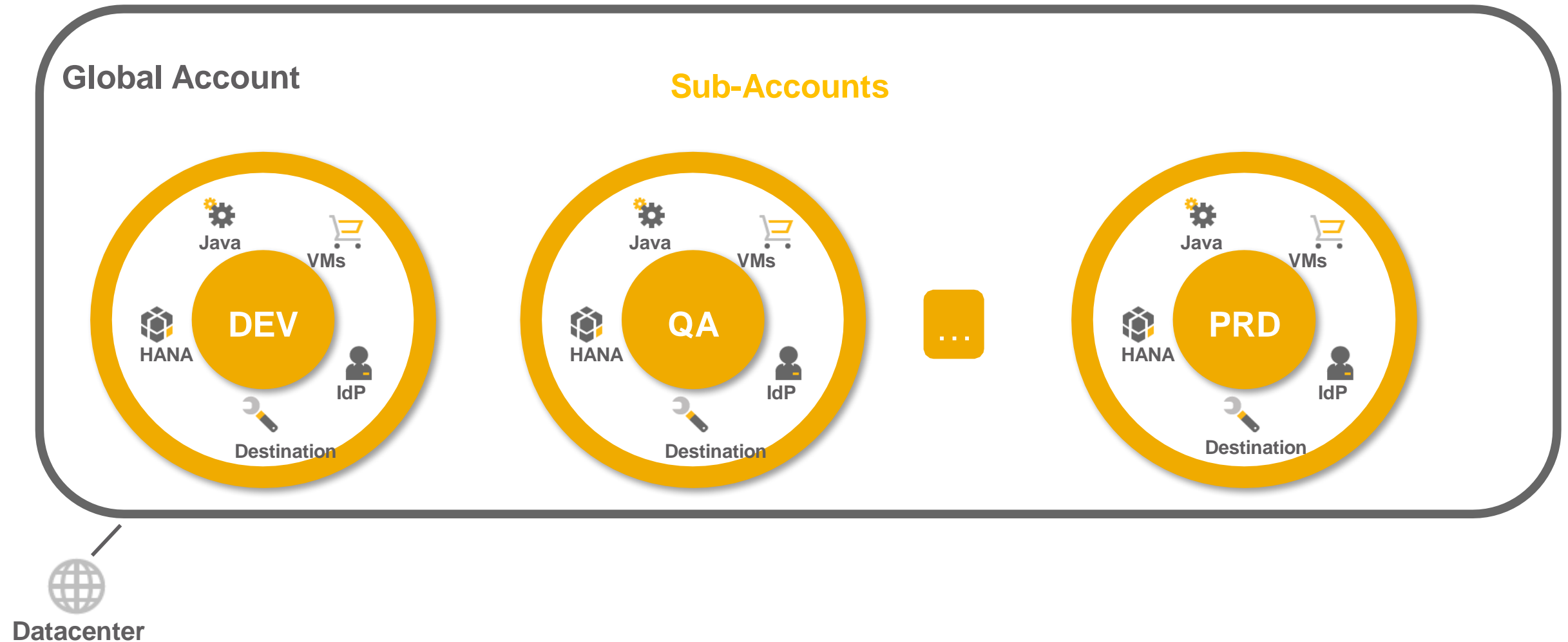
- **Resources** that can be consumed by apps
- **Users** allowed to work in the account
- **Apps** deployed and running in the account
- **Data** written by apps running in the account
- **Configuration** for apps running in the account

Each account is assigned to a **datacenter**. *

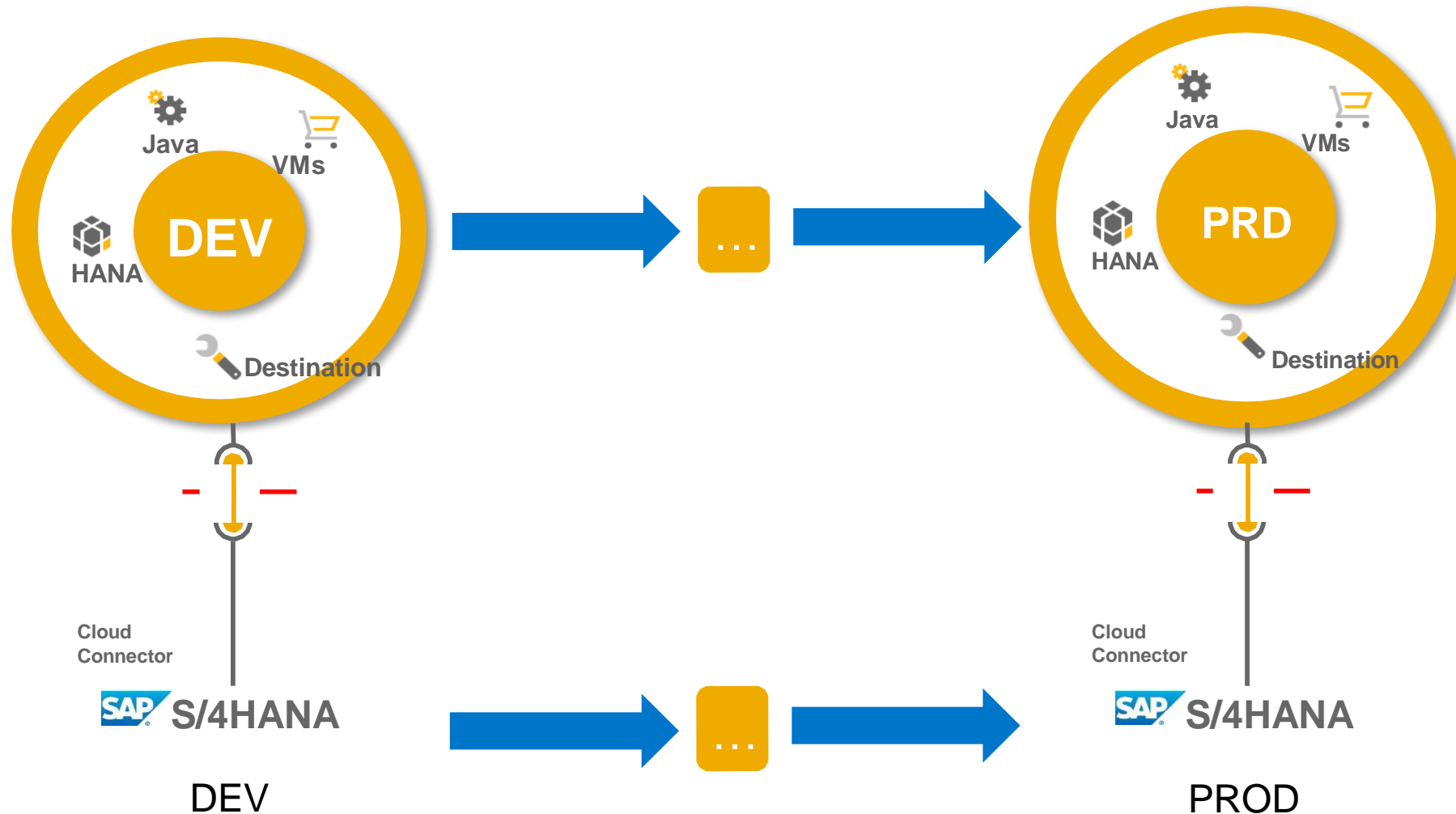
Each account is **fully isolated**.

*) New datacenters in setup

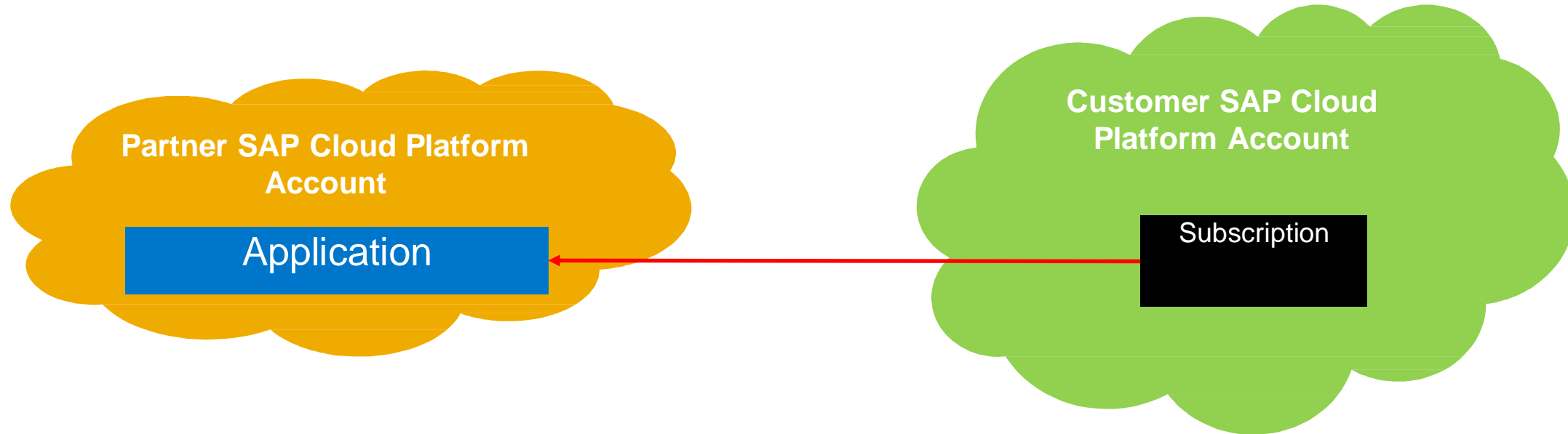
Sub – Accounts Concept



Pairing of Accounts with Backend Systems



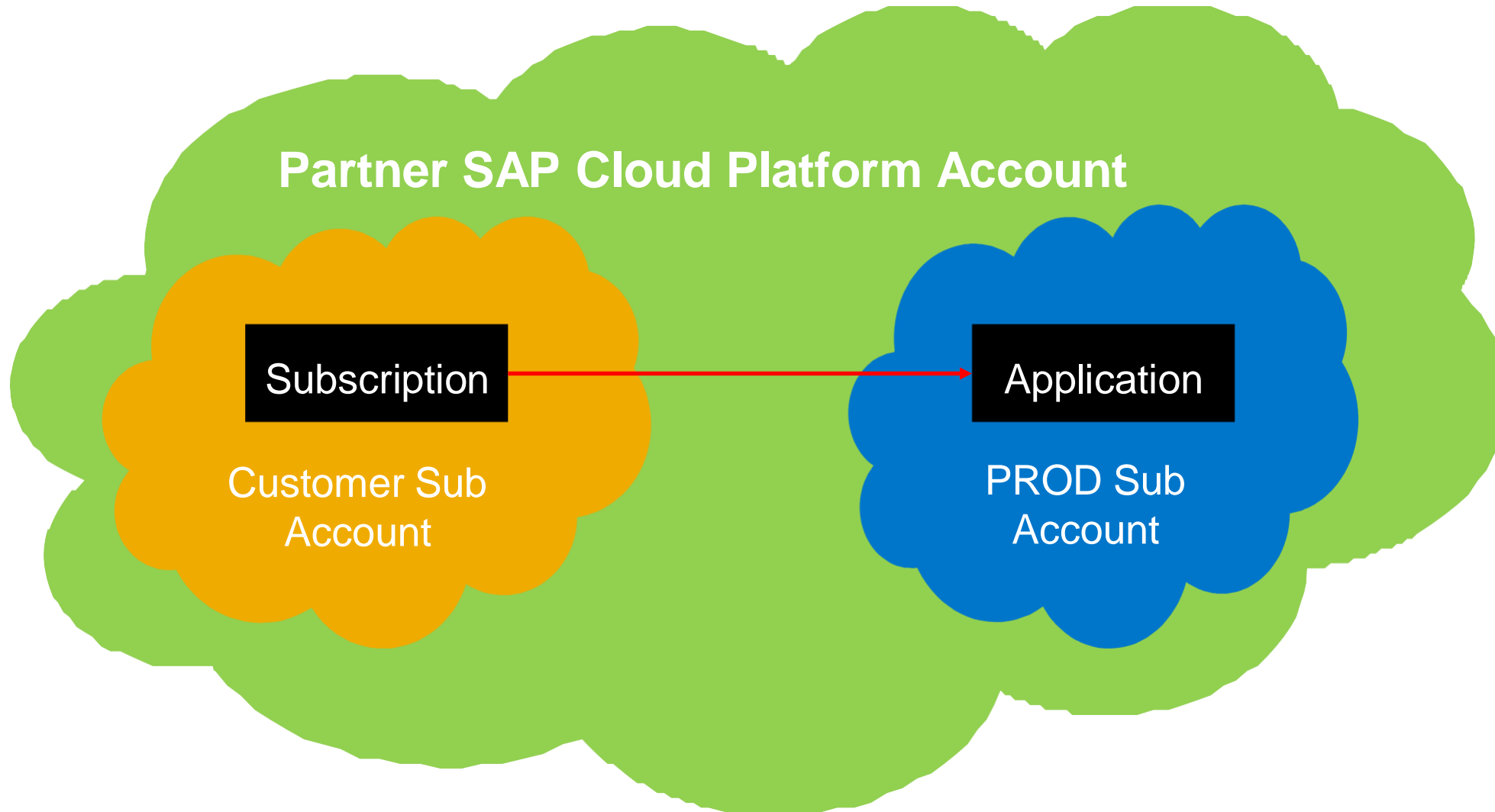
What is a Multi-Tenant application ?



Extension application resides on Partner account

Note: Extension application is not copied over to Customer account

Customer does not have an account...



General Programming Guidelines for multitenant programming

- Shared in-memory data such as Java static fields, caches, static tables will breach tenant isolation
- Avoid any possibility that an application user can execute custom code in the application JVM, as this **will certainly** give them access to other tenants' data
- Avoid any possibility that an application user can access a file system, as this **will certainly** give them access to other tenants' data.

Multitenant Core Services in HCP

- **Connectivity Service**
- **Identity and Access Management Service**
- **Persistency Service**
- **Document Service**
- **Keystore Service**
- **Extensions service**

Destination

The screenshot shows the 'SAP HANA Cloud Platform Cockpit' interface. On the left is a navigation menu with options: Dashboard, HANA XS Applications, Java Applications, Database Systems, Databases & Schemas, HTML5 Applications, Destinations (highlighted), Connectivity, Subscriptions, Services, Members, Useful Links, and Legal Information. The main area is titled 'Destination Configuration' and contains the following fields:

- Name: ABAP_GW_Backend
- Type: HTTP
- Description: (empty)
- URL: http://[redacted]
- Proxy Type: OnPremise
- Authentication: BasicAuthentication
- User: demo
- Password: (masked with dots)

Below these fields is the 'Additional Properties' section with two rows:

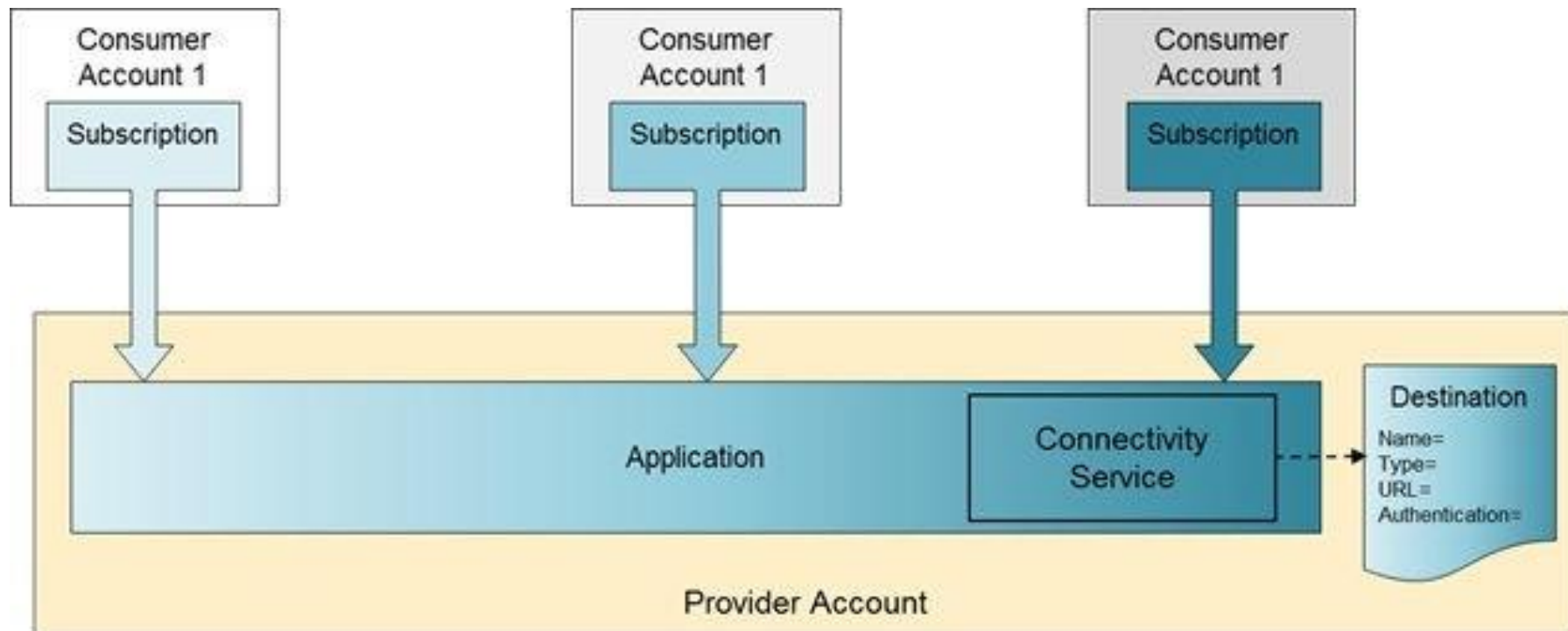
Property Name	Value
WebIDEEnabled	true
WebIDEUsage	odata_gen,odata_abap,ui5_execute_abap.

At the bottom are 'Save' and 'Cancel' buttons. A 'New Property' button is also visible next to the 'Additional Properties' table.

- Destinations are used for the outbound communication of a cloud application to a remote system
- Contain connection details for remote communication
- Contain symbolic names
- The connectivity service resolves the destination at runtime based on the symbolic name provided
- The currently supported destination types are **HTTP**, **Mail** and **RFC**.

Provider Specific Destination

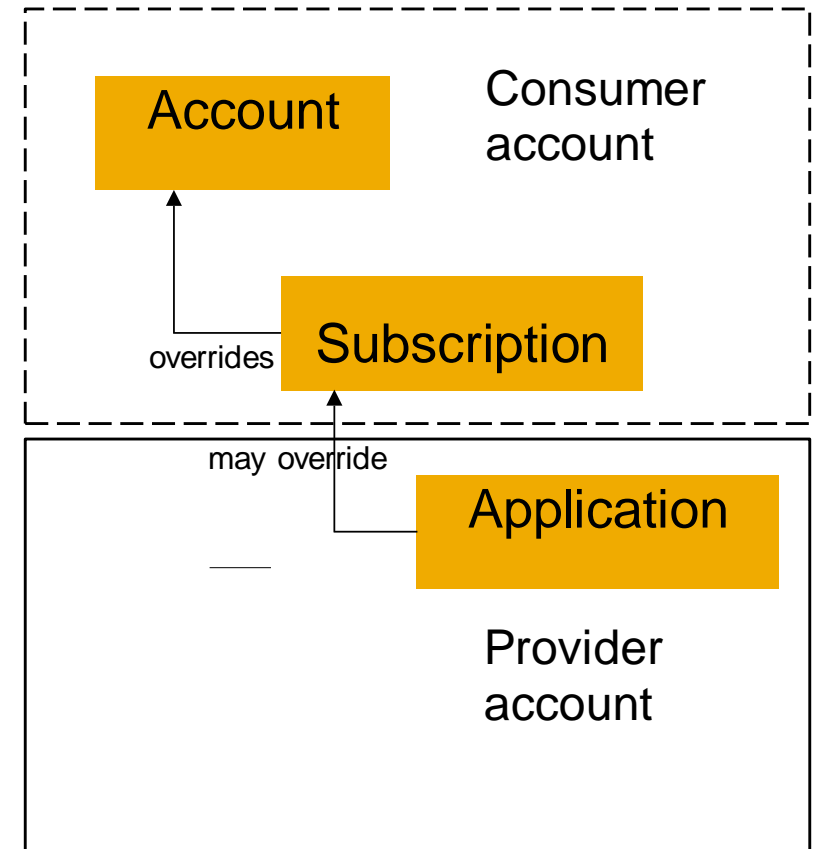
- A common shared destination (DestinationProvider=Application)
- The destination is always read from the provider account
- For example, weather data



Destination visibility

Destinations visibility according to the level:

1. Destination uploaded on account level - it is visible for the whole account
2. Destination uploaded on subscription level - it is only visible for the dedicated subscription
3. Destination uploaded on application level - it is visible by all tenants and accounts, regardless their permission settings

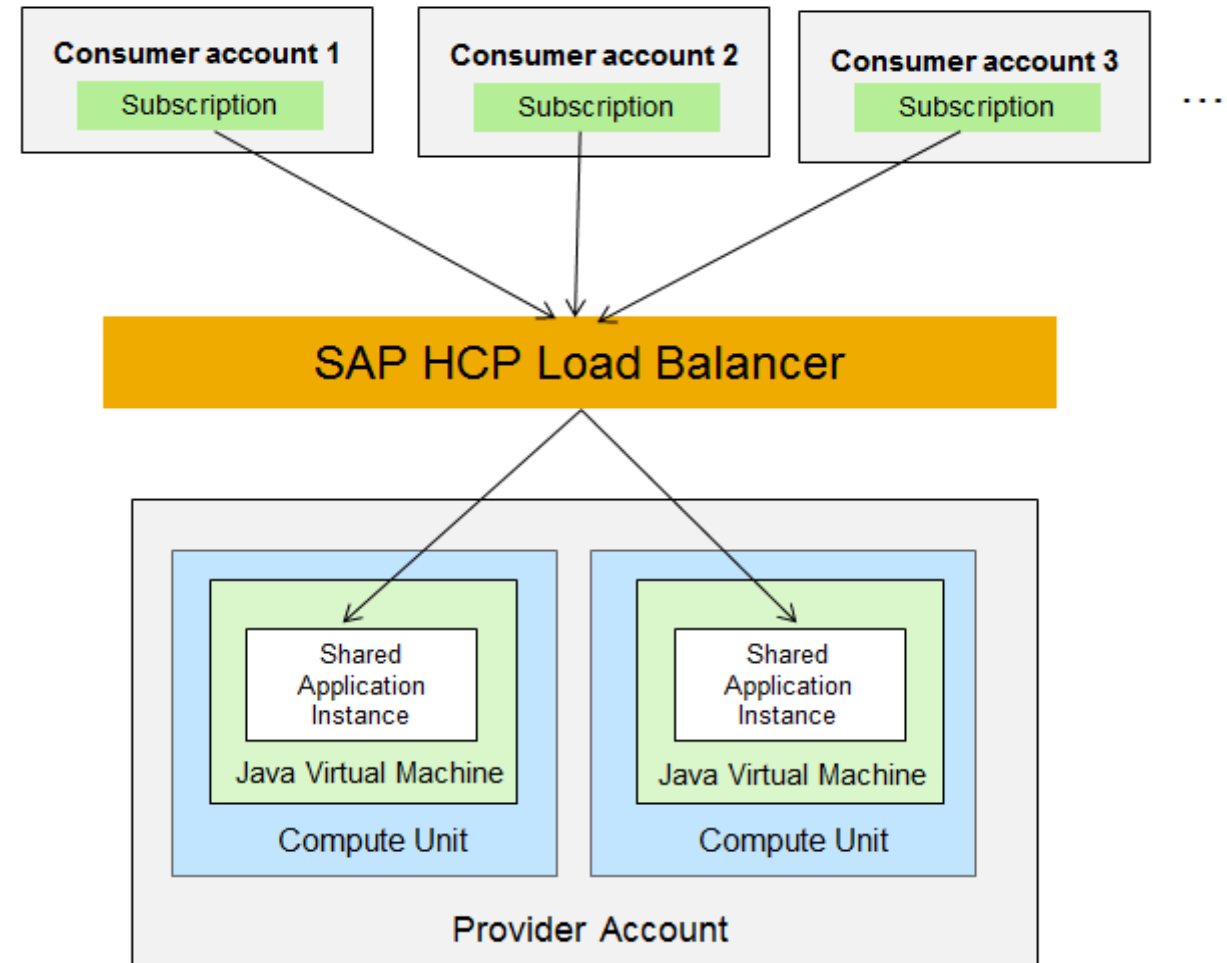


Scaling Applications

- Java Applications developed can run on one or more processes (VMs)
- Compute Unit quota for an account determines the number of processes that can be started
- One or more JVMs can be shared between all consumers in case of Multi-tenant applications
- Scaling down with soft shutdown on application processes

Advantages:

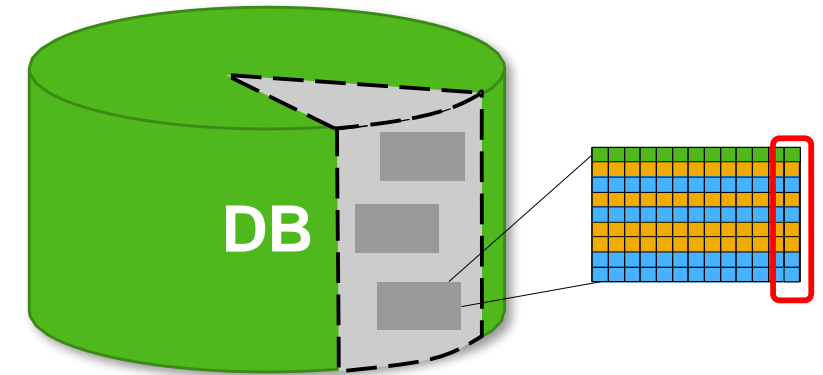
- Savings in resource costs
- Manage failover scenarios
- Savings in operational costs



Data Level Isolation using the Persistency service

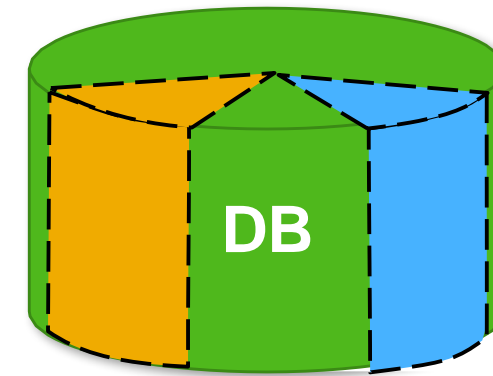
1. Sharing a single database schema between all application consumers

- a. Discriminatory field in each table
- b. Tenant ID can be used as a value in discriminatory field
- c. Each SQL statement should use the Tenant ID



2. Schema level isolation

- a. A schema for each tenant bound to the application
- b. The application uses JNDI to dynamically look up the data source

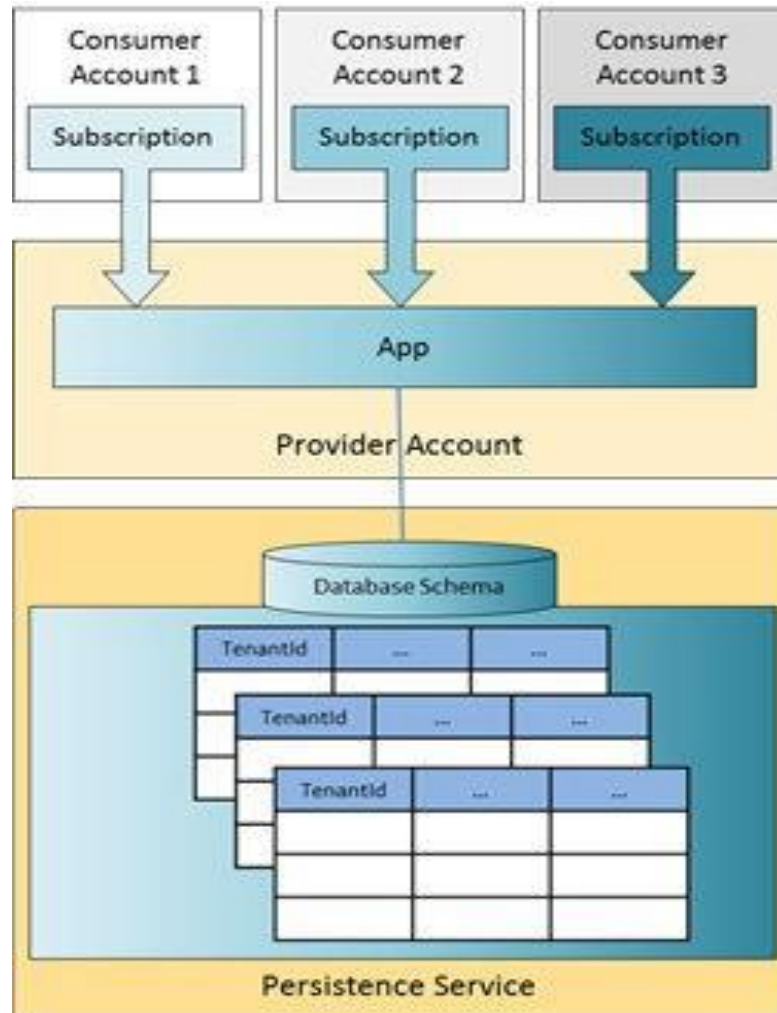


3. Multi-Data Container (MDC) isolation*

- a. A DB container for each tenant bound to the application
- b. The application uses JNDI to dynamically look up the data source

* Planned on HCP

Sharing a single database schema between all application consumers



- A single DB Schema is shared between all consumers
- Data separation done via discriminator column
- Can be easily implemented using EclipseLink / JPA
- Current tenant must be provided to entity manager

```
@Entity
@Table(name = "T_PERSON")
@Multitenant
@TenantDiscriminatorColumn(name="TenantId", contextProperty = "eclipselink.tenant-id", length=36)
@NamedQuery(name = "AllPersons", query = "select p from Person p")
public class Person {
```

```
HashMap<String, String> properties = new HashMap<String, String>();
properties.put(PersistenceUnitProperties.MULTITENANT_PROPERTY_DEFAULT, tenantId);
EntityManager em = emf.createEntityManager(properties);
```



Thank you

© 2014 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. Please see <http://global12.sap.com/corporate-en/legal/copyright/index.epx> for additional trademark information and notices.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors.

National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP SE or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP SE or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platform directions and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.