Salesforce
Using Edition-Based Redefinition

for Near-Zero-Downtime Releases

Lee Horner
DB Architect

Email: lhorner@salesforce.com
Twitter: @leehorner

Agenda

- Salesforce.com Overview
  - Enterprise Cloud Challenges at Massive Scale
- Implementing Edition-Based Redefinition (EBR)
- EBR Phased Rollout
- Planned Maintenance Benefits

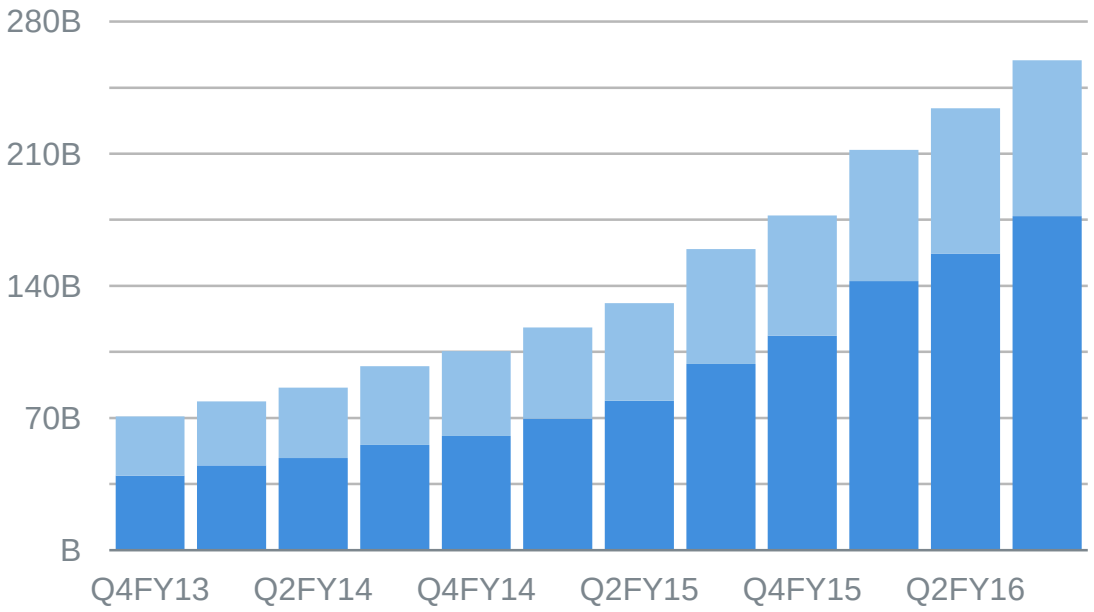
Salesforce Overview

Delivering Enterprise Cloud Transactions at Scale

Further Improving Response Times

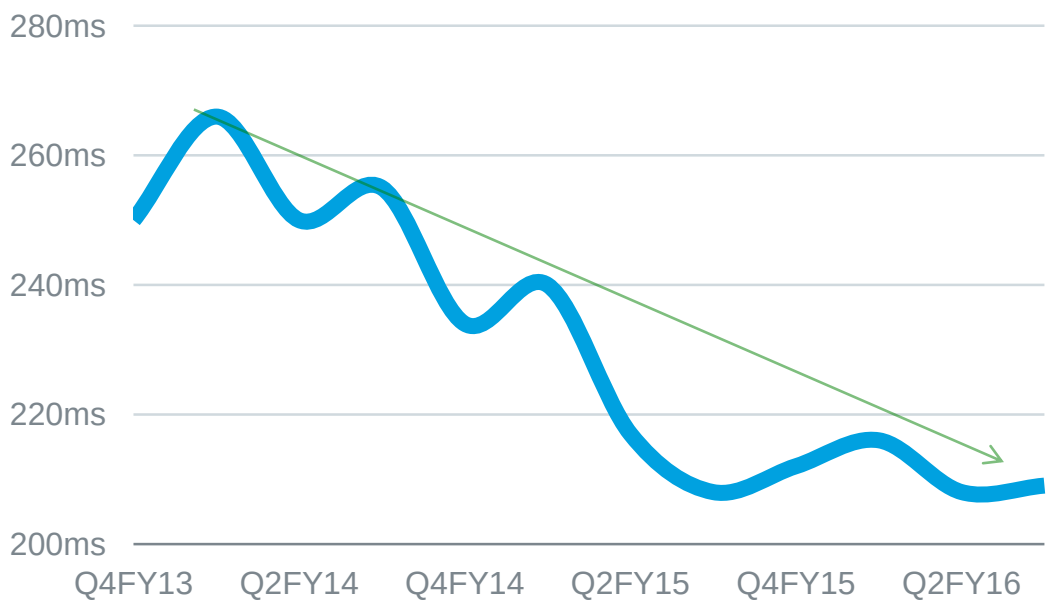
Transactions Per Quarter

259B Transactions in Q3FY16
63% YoY Growth (Q3FY15-Q3FY16)



Average Page Time

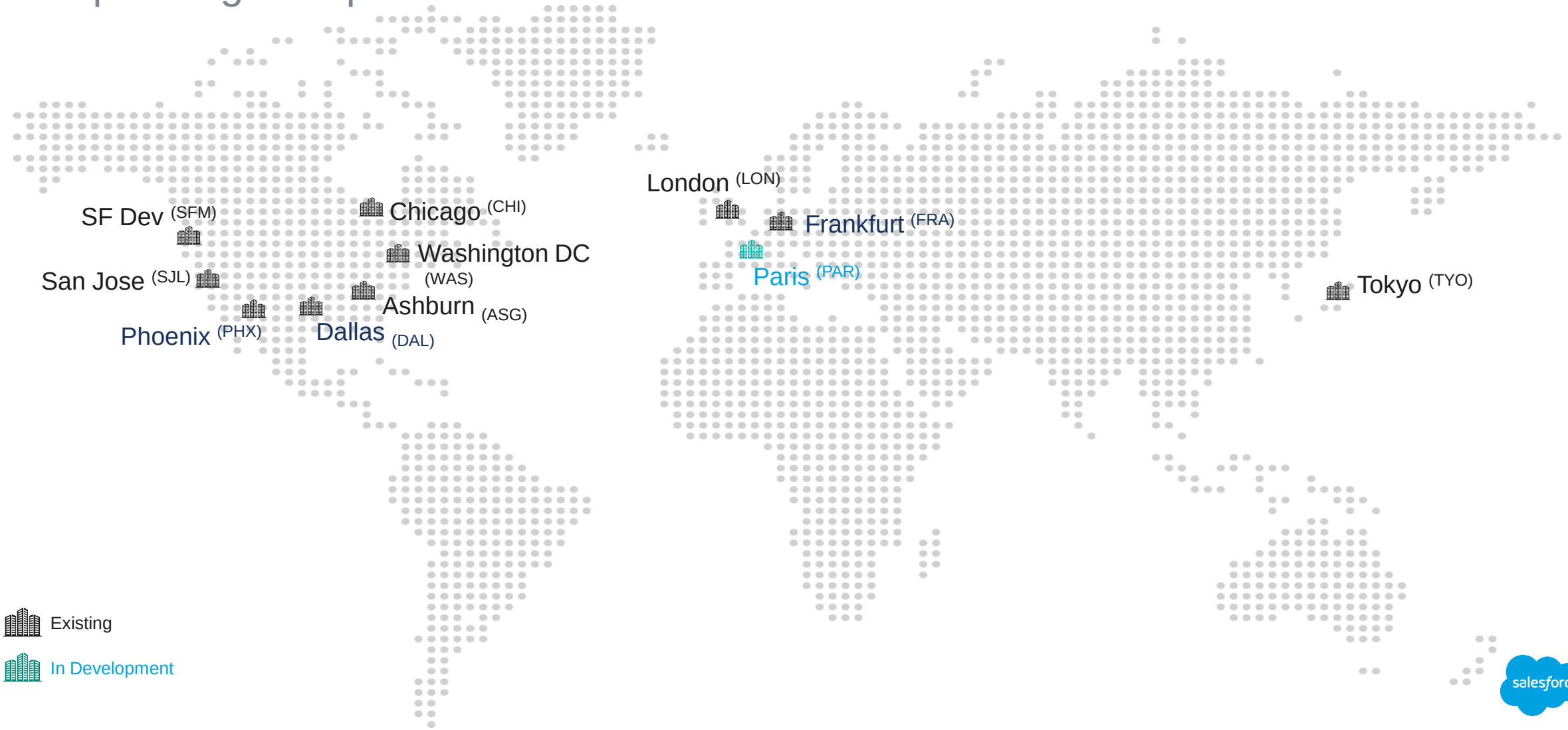
209ms Latency in Q3FY16
Flat (Reduction) YoY



salesforce

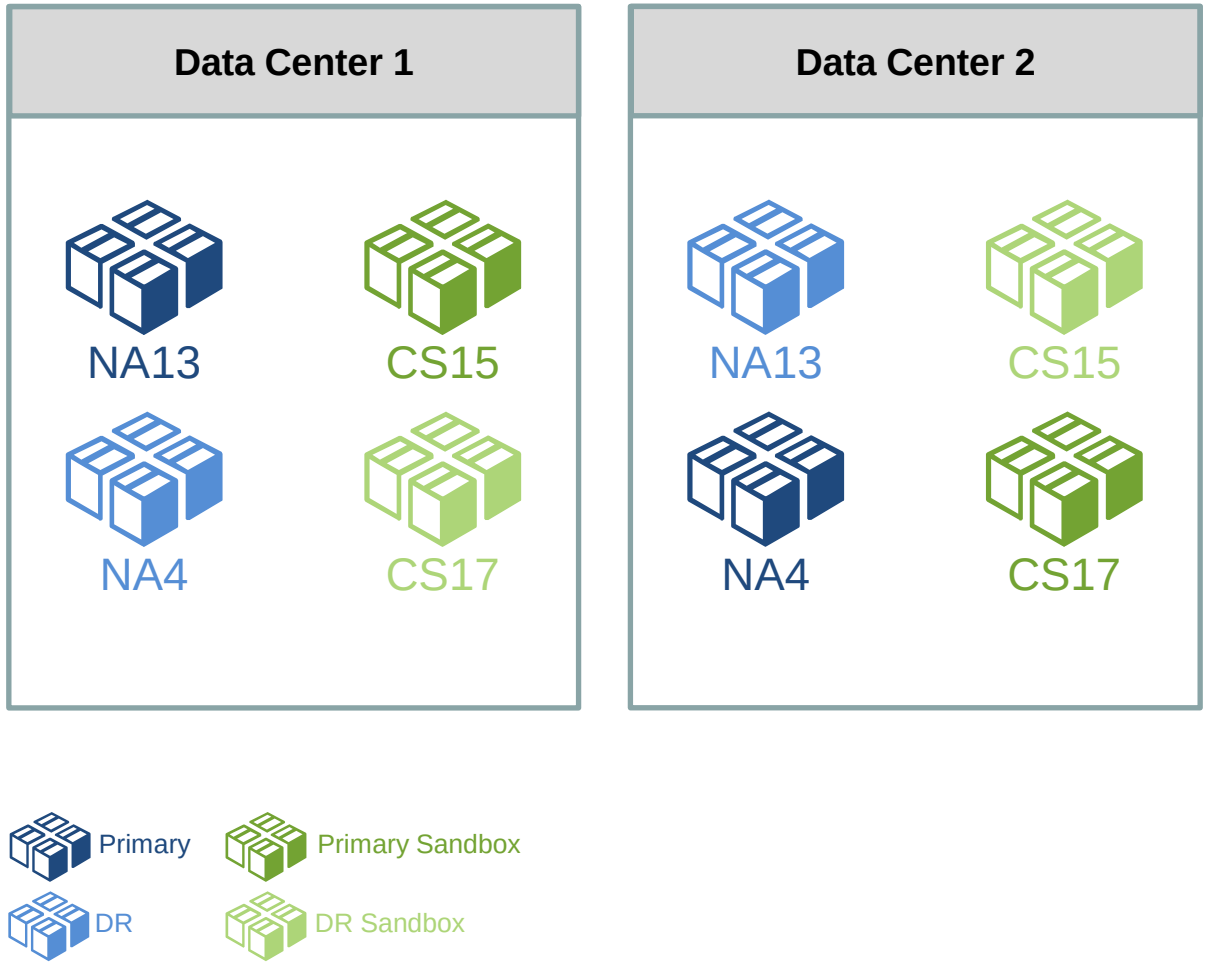
Global Enterprise Cloud Data Centers

Expanding Multiple Sites Worldwide



Designed for Enterprise Scalability and Reliability

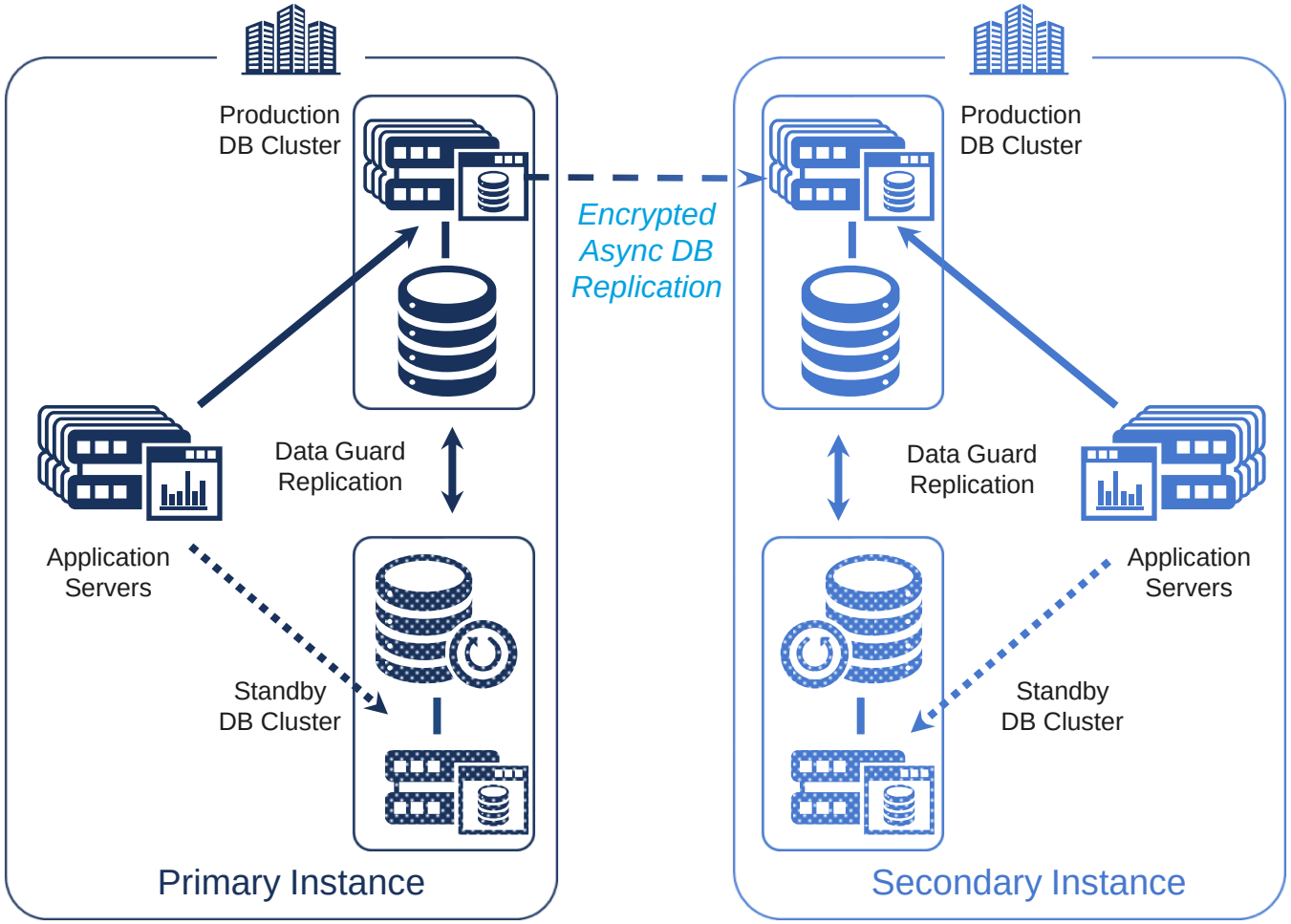
Disaster Recovery is Key Design Requirement



- We scale horizontally through multiple production Salesforce instances
  - 47 Core
  - 47 Sandbox (used for Customer UAT)
- **Every** core production (NA, EU, AP) instance has an exact replica (mirror) in a second data center
- **Every** sandbox (CS) instance also has an exact replica in a second data center

Tremendous Focus on Core DB Reliability

While also ensuring high performance



Very large average DB sizes

- Core DB: **35+TB**
- Sandbox DB: **50+TB**

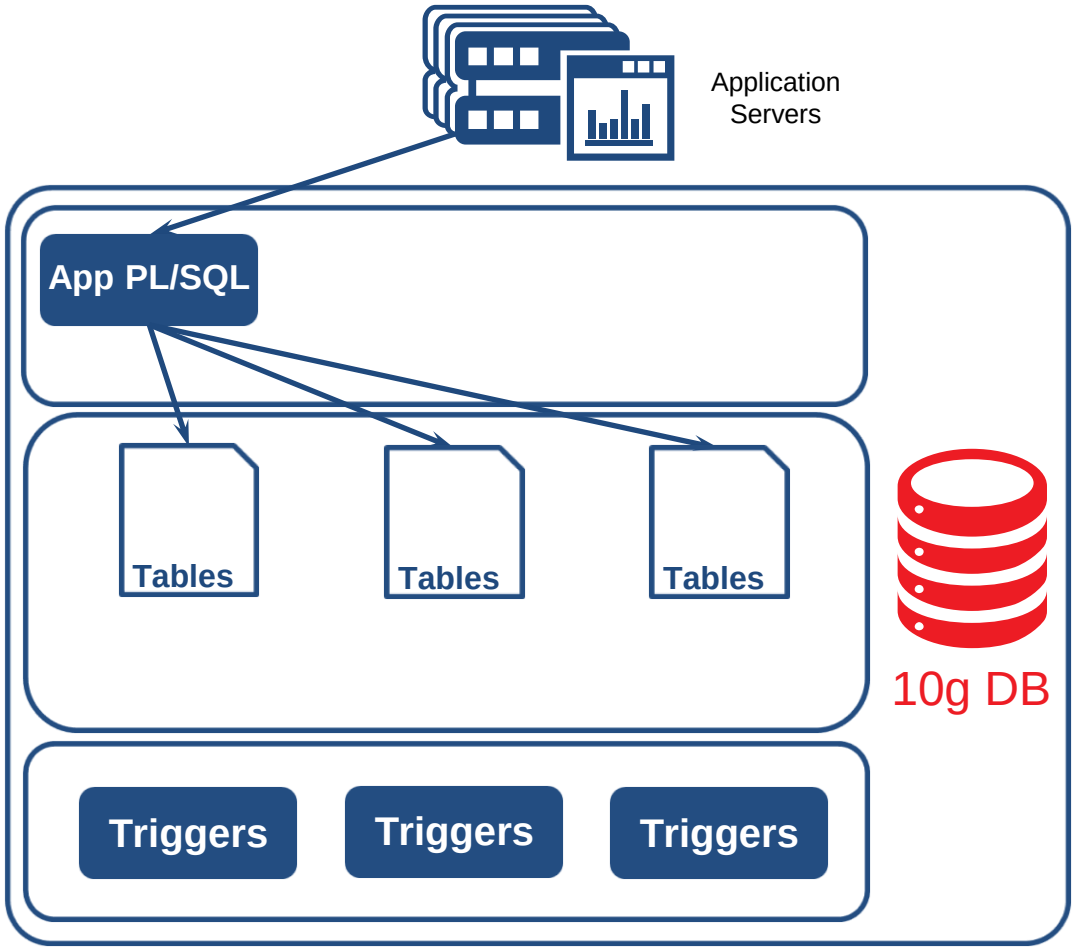
Scale vertically through RAC Clusters

Heavy use of Data Guard

Overview of Oracle DB at Salesforce.com

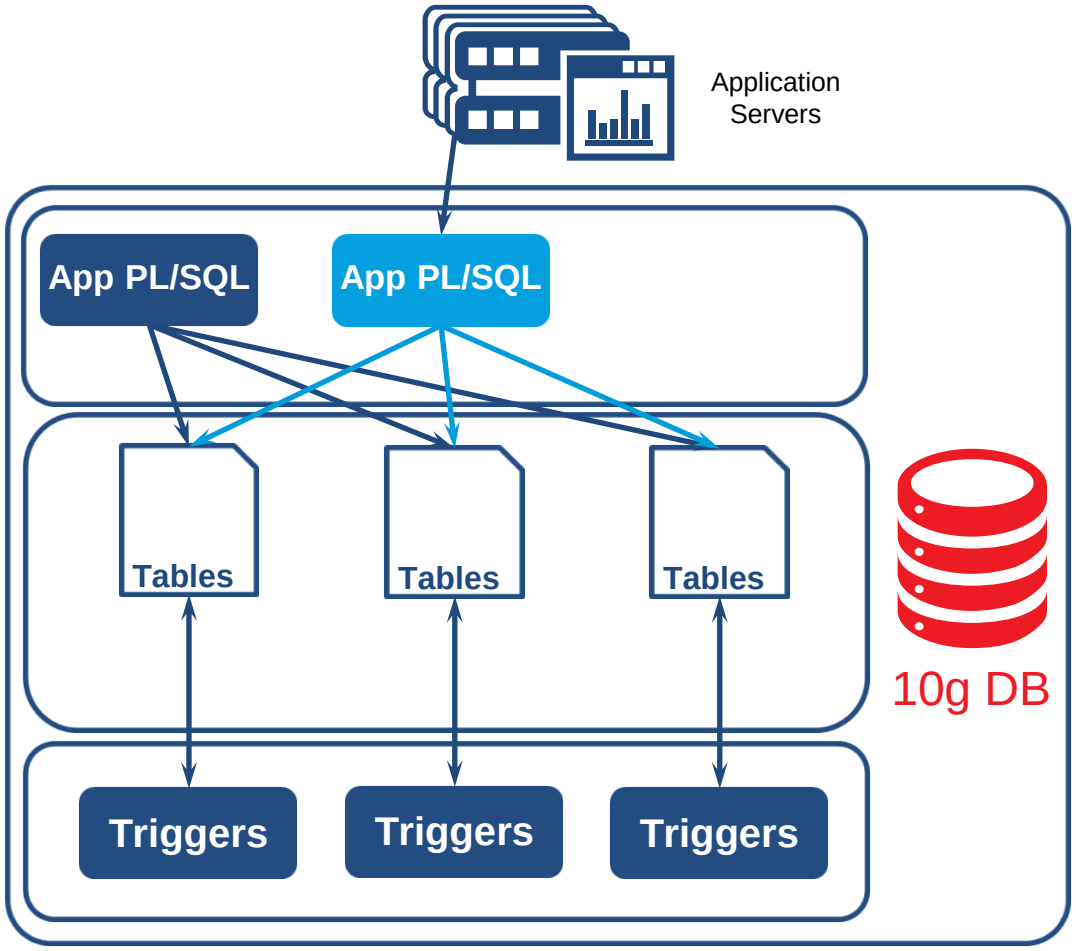
- We have classes of schemas for PL/SQL and tables, tables and PL/SQL do not co-exist inside schemas
- Metadata enables customisation of each customer's view of the service
  - Implications: requires large shared pools
- Use of multitenancy inside our tables
  - Implemented our own version of "multi-tenancy"
  - Each DB contains one version of each app table
  - Each row has a unique customer id
  - Careful security measures are in place to ensure customers see only their data
  - More details see https://developer.salesforce.com/page/Multi_Tenant_Architecture

Salesforce Application Upgrades Before 2010



- Major releases used to take 6 hours per release
  - Minimal use of “upgrade time” triggers
  - Large “batch” DML operations inside the planned maintenance window
- We knew we could do better
- Redesigned the upgrade process and moved to a technique of triggers that are aware of the major release version and upgrade only DB triggers

Salesforce Application Upgrades in 2010



- Major releases require many DDL operations to introduce new tables, columns, index changes
- Deliver 3 major releases and many minor update releases per year
- Minimal downtime for major releases
  - 30 minute “upgrade pre-release scripts”
  - With <5 minute restart
- No downtime for minor update releases

salesforce

Application Upgrades at Salesforce

- Each new major release has it's own "release specific" trigger code
- Implemented through the use of package variables
- Each app server connection establishes context and sets the "release version"

```
CREATE OR REPLACE TRIGGER app_trigger.t_v24_accounts
AFTER UPDATE
ON app_data.accounts
BEGIN
    IF app_plsql.global_context.getAppVersion = app_plsql.release.release_version.RELEASE_24 THEN
        ….;
    END IF;
END t_v24_accounts;
/
```

Application Upgrades at Salesforce

- This technique reduced planned maintenance from 6 hours to 30 minutes per major release
- Why 30 minutes?
  - Quiesce the database and the app servers
  - Run DDL and recompile PL/SQL
  - Restart app servers
- Could continue to iterate and improve our custom code to reduce this further
- Essentially attempting to do something very similar to EBR
- EBR will provide us with what we implemented but out of the box

Implementing EBR

Wait Didn’t We Invent That Already?

What is Edition-Based Redefinition

Oracle Definition* of EBR:

“Edition-based redefinition enables you to upgrade the database component of an application while it is in use, thereby minimizing or eliminating down time.”

* http://docs.oracle.com/cd/E11882_01/appdev.112/e10471/adfns_editions.htm#ADFNS020

Application Upgrades at Salesforce

- We “editions enabled” our schemas containing tables and PL/SQL
- We created editioning views against all tables
- We can load new versions on our PL/SQL code inside new editions
- We can perform DDL on tables without invalidating PL/SQL

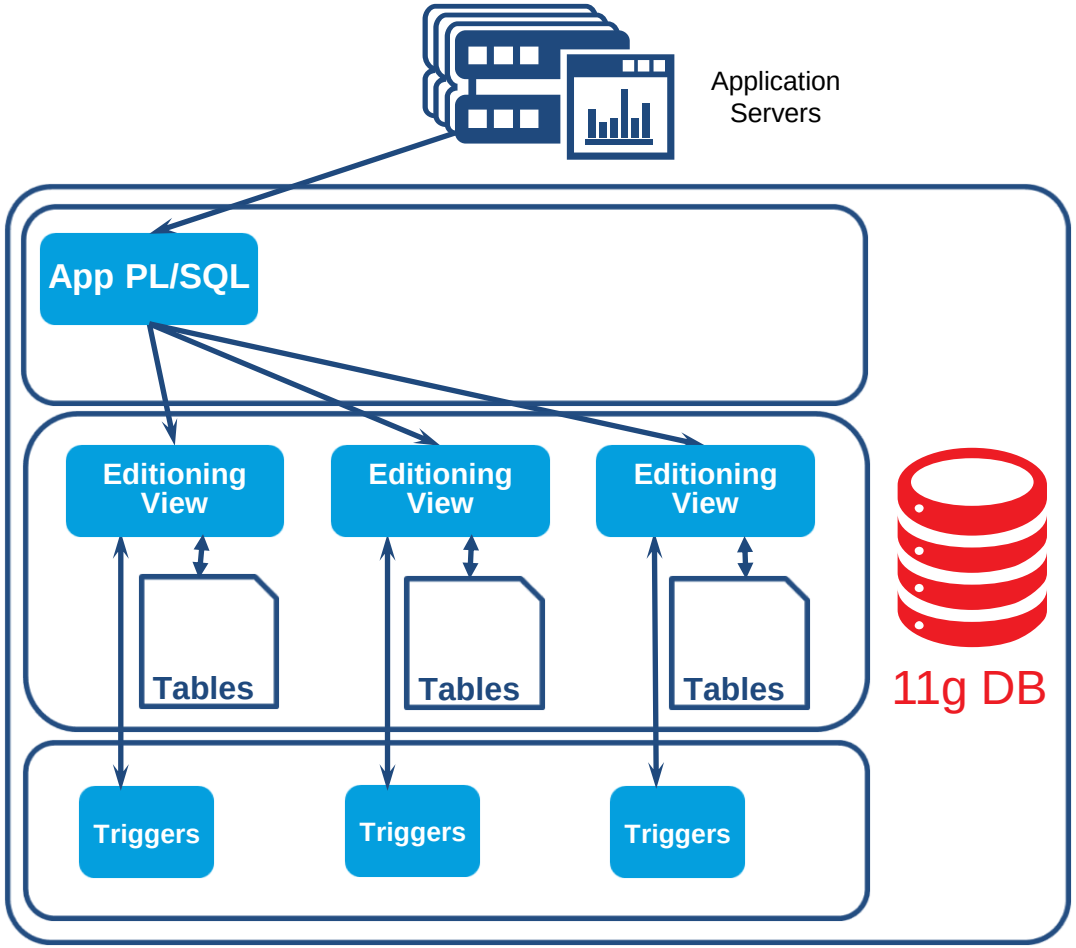
Salesforce Implementation of EBR

Following Oracle Recommended Best Practices

- How are we implementing EBR
  - All schemas containing data are editions enabled
  - All application PL/SQL schemas are editions enabled
  - All application schemas refer to editioning views inside our schemas containing data
  - No application schemas have any grants on any tables

Salesforce Implementation of EBR

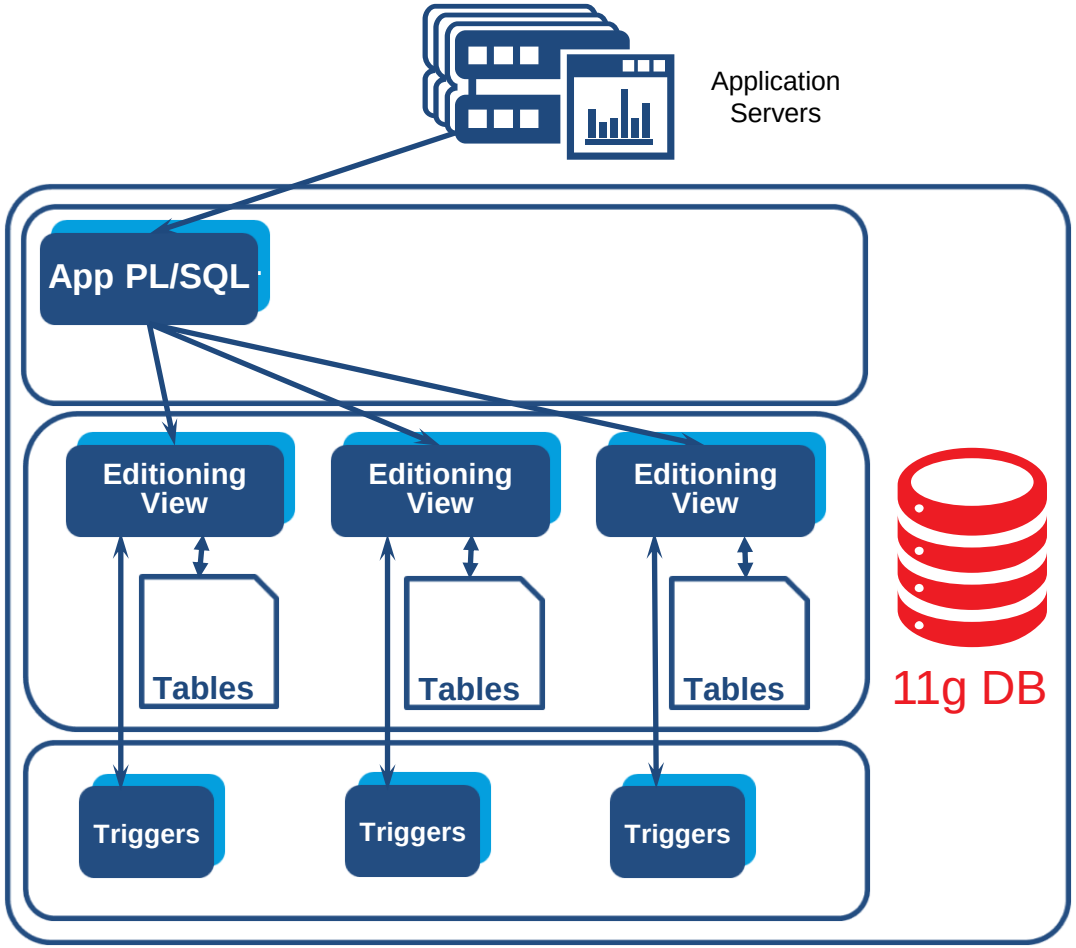
EBR Running Current Edition



- All schemas are editioned
- All Application PL/SQL code no longer refers to tables directly
- All tables have a corresponding editioning view

Salesforce Implementation of EBR

EBR Running New Edition



- A new release is made by creating a new edition
- Upgrade time triggers can be implemented as cross edition triggers
- When moving to a new release we change the running edition

EBR Phased Rollout

Rolling out EBR at Salesforce

Taking a Phased Approach

- We are conservative when rolling out significant changes
- Initial rollout to two production environments (GUS and 1 Sandbox)
- Remaining instances upgraded in subsequent release

Rolling out EBR at Salesforce

Taking a Phased Approach

Maintenance Calendar ?

Monthly All | Calendar List

Instance(s)	Name	Start (UTC)	End (UTC)	Message
NA0, NA6, NA21, NA29	Database Maintenance	Jan 17 2016, 05:00	Jan 17 2016, 05:45	This instance will not be available during this maintenance window.
NA0, NA3, NA16, NA21, NA32	Spring '16 Major Release	Feb 13 2016, 04:00	Feb 13 2016, 04:05	This instance will not be available during this maintenance window.
NA0, NA3, NA16, NA21, NA32	Summer '16 Major Release	Jun 11 2016, 03:00	Jun 11 2016, 03:05	This instance will not be available during this maintenance window.
NA0, NA3, NA16, NA21, NA32	Winter '17 Major Release	Oct 15 2016, 03:00	Oct 15 2016, 03:05	This instance will not be available during this maintenance window.

Dates and times subject to change. **Safe harbor.**

EBR will be implemented in the 30 minute planned maintenance window of a major release

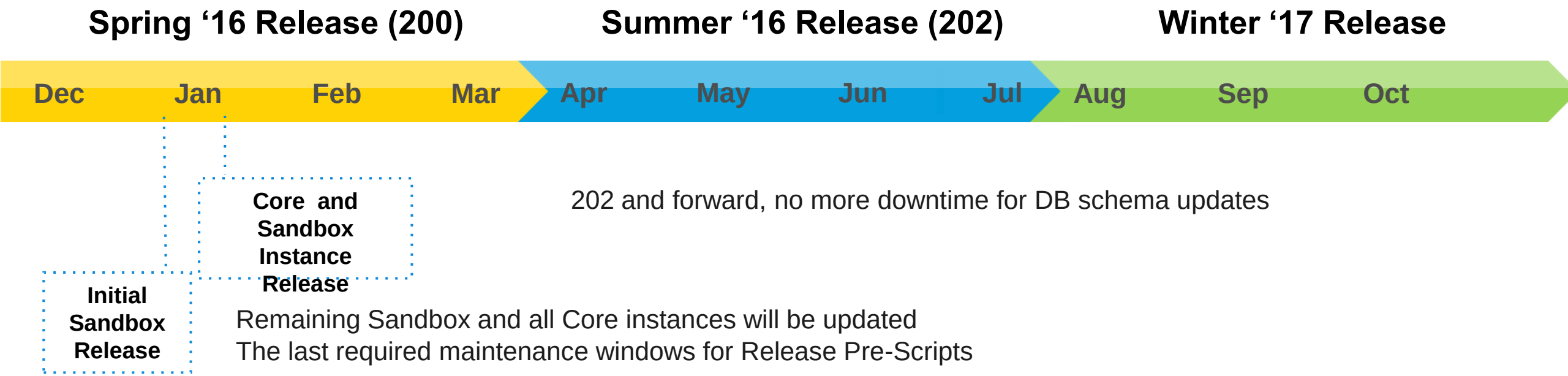
EBR is implemented with no new maintenance for our customers

EBR is implemented to minimize application changes

salesforce

Phased EBR Releases

Ensures High-Quality Deployments



The Rollout of EBR at Salesforce

- Steps to implement EBR
  - Editions enable all schemas containing data and application PL/SQL
  - Revoke all access to base tables
  - Rename all base tables
  - Drop all triggers on base tables
  - Build editioning views on base tables using the original name
  - Grant access on all editioning views to PL/SQL schemas
  - Recreate all triggers (against editioning views)
  - Recompile all PL/SQL
  - Almost all of this generated through data dictionary queries

Impact of Phased Rollout of EBR

A Staggered Rollout Will Have Implications

- For a release we effectively doubled our requirements for automated testing and performance testing
- Must ensure the correctness of both the EBR and the non-EBR version of the service
- Easier to test against an unchanging DB (offline), trigger-based version also increased the testing requirements
- This is more work for developers working on upgrade code
- The elapsed time of the planned maintenance will be completely predictable

Planned DB Maintenance Benefits

Benefits of EBR at Salesforce

- Reduces planned maintenance for major releases
  - The 11gR2 fine grained dependency model means no PL/SQL/editioning view invalidations
  - More non-blocking DDL inside 11gR2 such as ALTER TABLE ADD COLUMN
- When we reduced the downtime from 6 hours to 30 minutes it increased the number of triggers required
- We now have two versions of triggers on the table
  - One for release N and another for release N+1
- With EBR we place these triggers in separate editions
- There will be a performance improvement with the reduction of triggers

thank you