ORACLE®

hroug
hrvatska udruga oracle korisnika

Online Application Upgrade with Edition Based Redefinition

Morana Kobal Butković, Senior Sales Consultant
Oracle Hrvatska

Oracle Database 11g Release 2

Online Application Upgrade

- Large, mission critical applications are often unavailable for tens of hours while a patch or an upgrade is installed
- Oracle Database 11g Release 2 introduces revolutionary new capabilities that allow online **application** upgrade with uninterrupted availability
- The pre-upgrade application and the post-upgrade application can be used concurrently

Online Application Upgrade

Requirements

- The installation of the upgrade into the production database must not perturb live users of the pre-upgrade application
  - Many objects must be changed in concert. The changes must be made in privacy
- Transactions done by the users of the pre-upgrade application must by reflected in the post-upgrade application
- For hot rollover, we also need the reverse of this:
  - Transactions done by the users of the post-upgrade application must by reflected in the pre-upgrade application

The solution

Edition-based redefinition

- Revolutionary new features:
  - edition
  - editioning view
  - crossedition trigger
- Code changes are installed in the privacy of a new edition
- Data changes are made safely by writing only to new columns or new tables not seen by the old edition
  - An editioning view exposes a different projection of a table into each edition to allow each to see just its own columns
  - A crossedition trigger propagates data changes made by the old edition into the new edition's columns, or (in hot-rollover) vice-versa

The challenge

Application versioning

- Scenario
  - The application has 1,000 mutually dependent tables, views, PL/SQL units, and other objects
  - There's more than one schema
  - They refer to each other by name – and often by schema-qualified name
  - The upgrade needs to change 10 of these

The challenge

Application versioning

- Can’t change the 10 objects in place because the other 990 refer to them – and this would require changing the pre-upgrade application
- Through 11.1 object determined by name and owner
- naming mechanisms through 11.1 are not rich enough to support online application upgrade

The solution

Editions

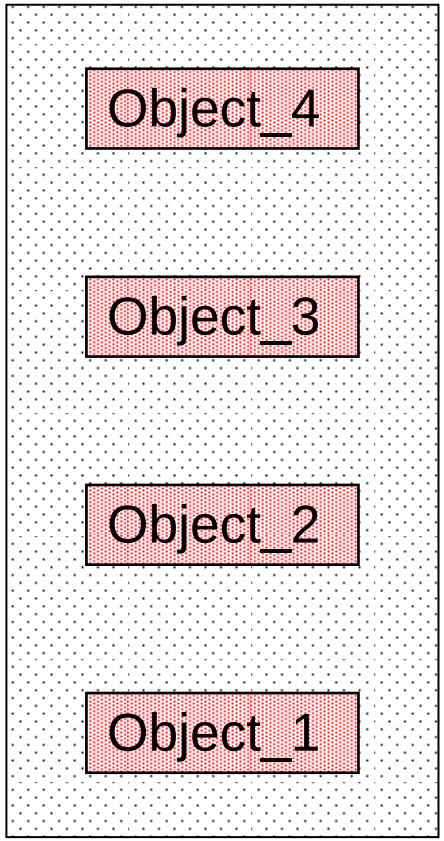
- Edition - new nonschema object type in 11.2
- An edition is the child of an existing edition
  - An edition inherits all objects from its parent
  - Each edition can have its own private occurrence of “the same” object
  - An edition can’t have more than one child
- A database must have at least one edition
- A database session always has a current edition

The solution

Editions

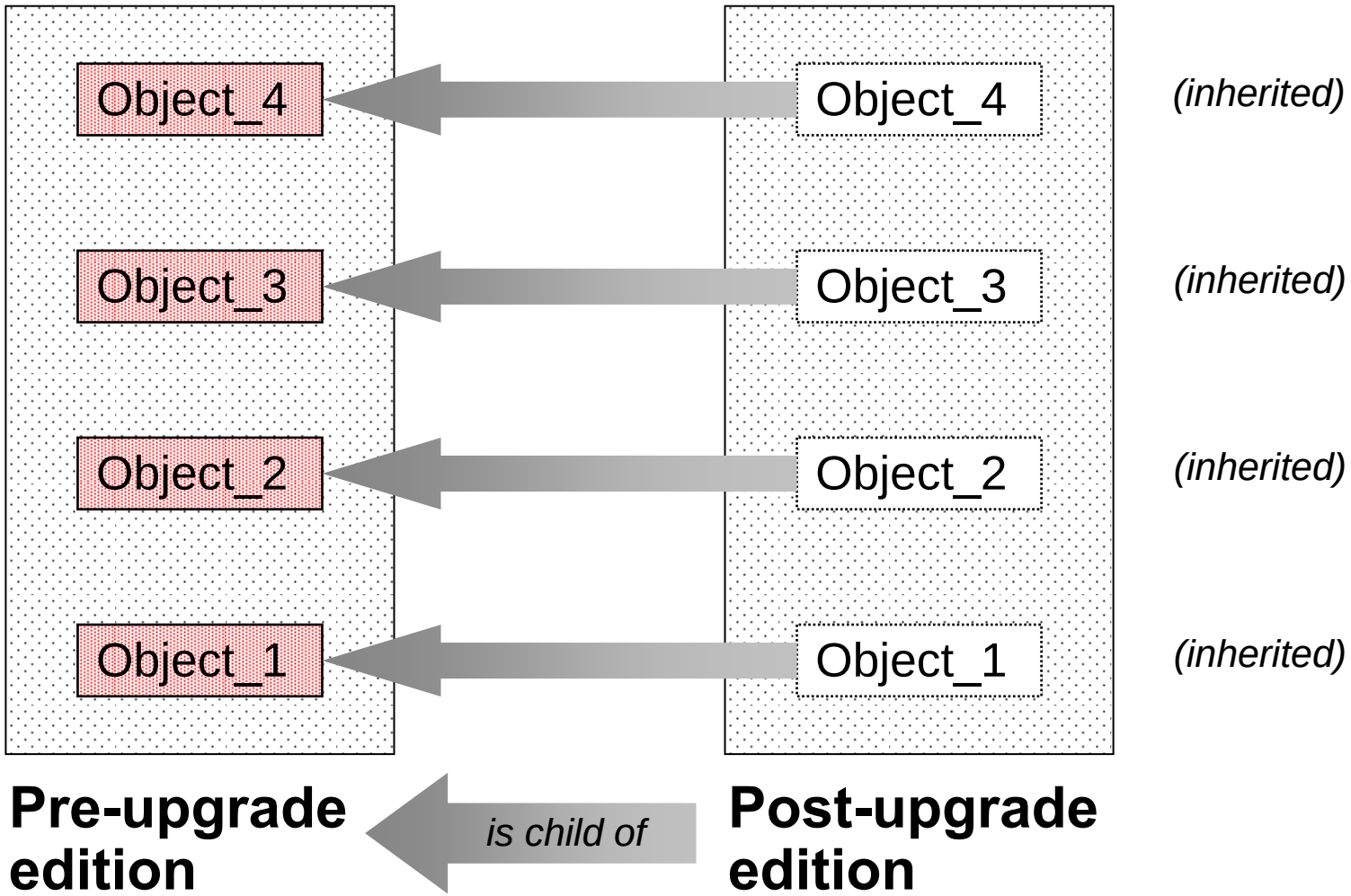
- From 11.2 editioned objects identified by name, owner and the edition in which it was created
- Name and owner are interpreted in the context of a current edition of the current session
- Visibility rules: find the object with the given name and owner in the closest ancestor edition

Editions & object visibility

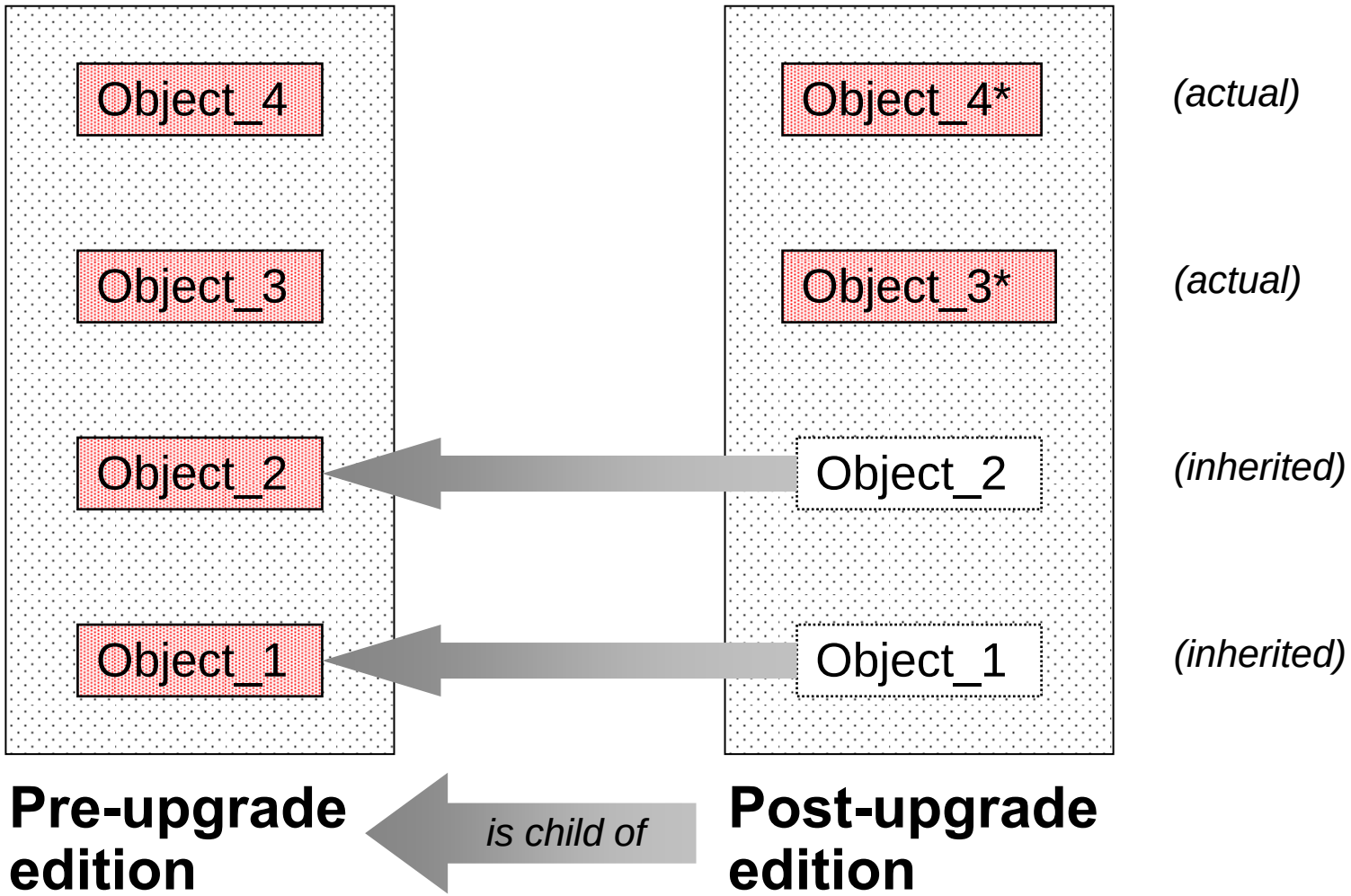


Pre-upgrade edition

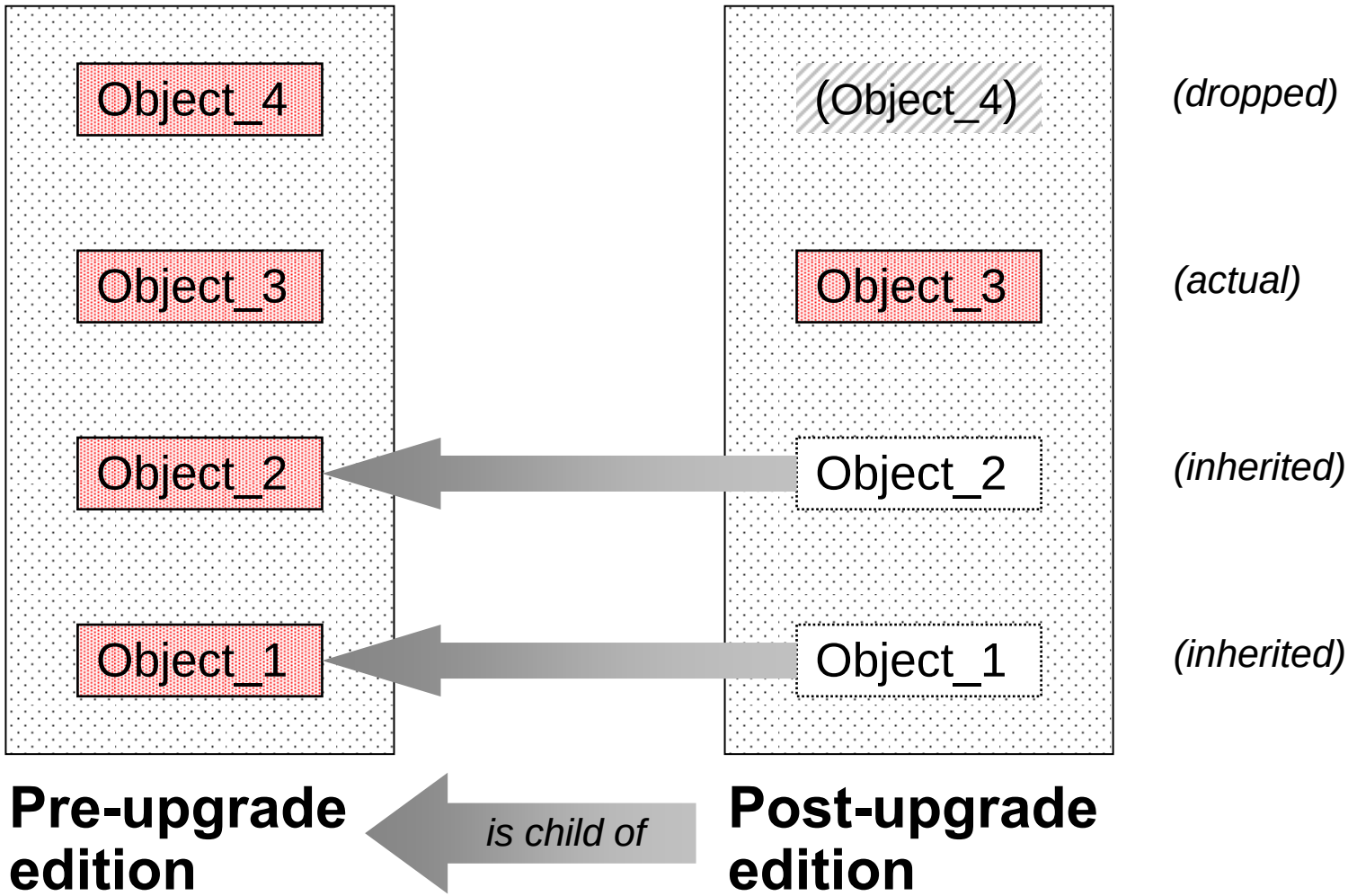
Editions & object visibility



Editions & object visibility



Editions & object visibility



Editions

- If your upgrade needs only to change synonyms, views, or PL/SQL units, you now have all the tools you need

Editionable and noneditionable object types

- Not all object types are editionable
  - Synonyms, views, and PL/SQL units of all kinds (including triggers and libraries) are editionable
  - Objects of all other object types e.g. tables are noneditionable
- Version the structure of a table manually
  - Instead of changing a column, you add a replacement column
  - Then you rely on the fact that a view is editionable

Editioning views – dealing with data across an Application upgrade

- An editioning view may only project and rename columns
- The EV must be owned by the table's owner
- Application code should refer only to the logical world
- Like all views, an editioning view can be read-only
- If you can tolerate only read access to the underlying data for an editioning view that the upgrade will change, you now have all the tools you need

What if data is changing during upgrade?

- If schema changes are required then both old and new structures must be must be maintained during upgrade process
- Triggers have the ideal properties to do this safely
- Each trigger must fire appropriately to propagate changes to pre-upgrade columns into the post-upgrade columns – and vice versa

The solution

Crossedition triggers

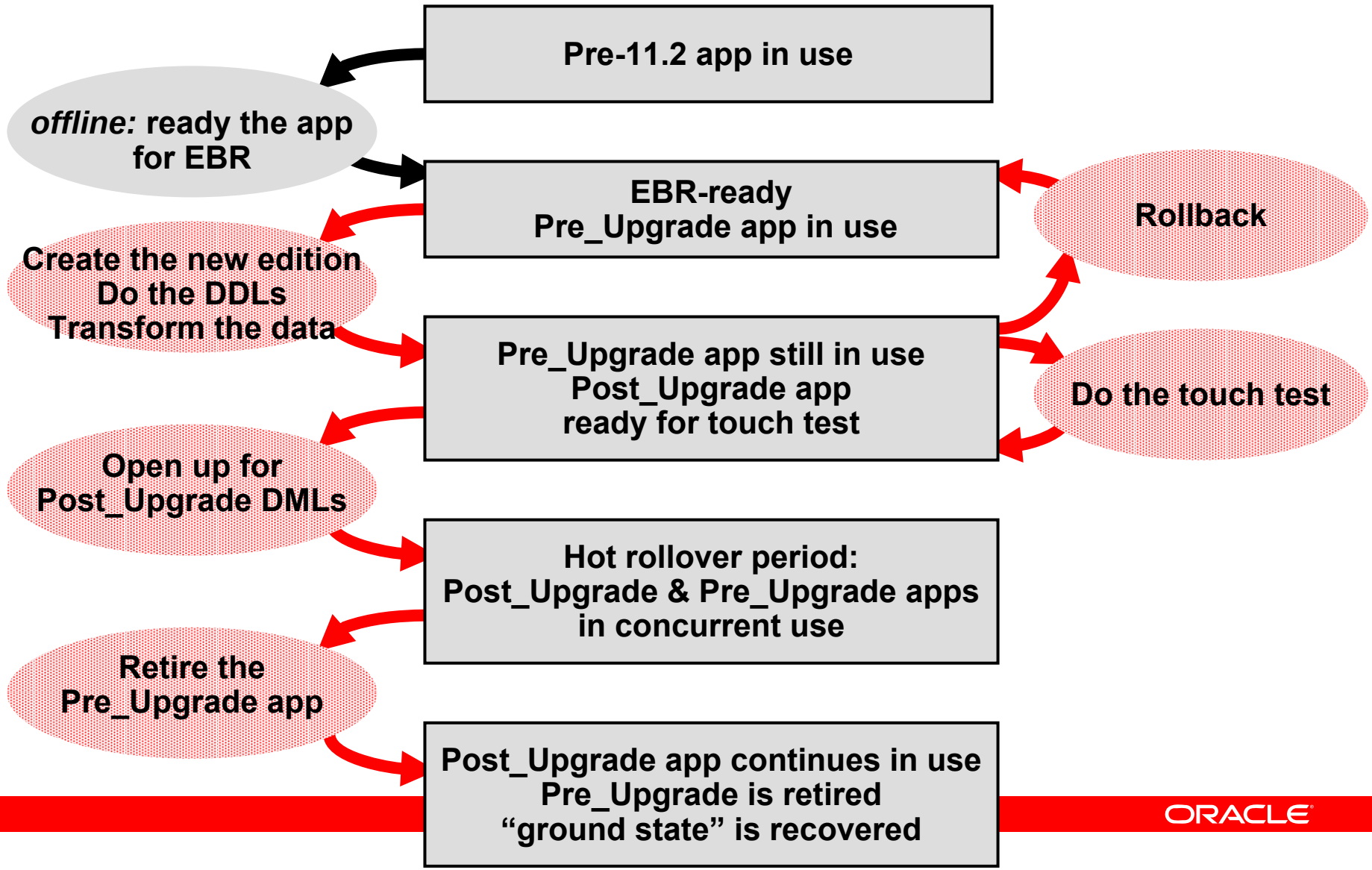
- Crossedition triggers directly access the table.
- The new crossedition trigger has special firing rules
- You create crossedition triggers in the Post_Upgrade edition
- The firing rules rules assume that
  - Pre-upgrade columns are changed – by ordinary application code – only by sessions using the Pre_Upgrade edition
  - Post-upgrade columns are changed only by sessions using the Post_Upgrade edition

The solution

Crossedition triggers

- A forward crossedition trigger is fired by application DML issued by sessions using the Pre_Upgrade edition
- A reverse crossedition trigger is fired by application DML issued by sessions using the Post_Upgrade edition
- The SQL that a crossedition trigger issues always executes in the edition that owns it:
  the Post_Upgrade edition

EBR task flow – summary



Case study

- The HR sample schema, as shipped by Oracle Corp, has a single column for phone number
  - Diana Lorentz 590.423.5567
  - John Russell 011.44.1344.429268
- It must be split into two new columns
  - Country code
  - Number within country
  - Diana Lorentz + 1-590-423-5567
  - John Russell + 44-134-442-9268

DEMONSTRATION

Edition-based Redefinition

Q&A