Application Express with Oracle Database 12c Multitenant Architecture
The following is intended to outline Oracle’s 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 remain at the sole discretion of Oracle.
Database Cloud & Consolidation Architectures

**Virtual Machines**

Share servers with “VM isolation”

**Multiple DB Instances**

Share servers and OS with “DB Instance Isolation”

**Schema Consolidation**

Share servers, OS and database with “Schema Isolation”
Database Cloud & Consolidation Architectures

**Virtual Machines**
Share servers with "VM isolation"

**Multiple DB Instances**
Share servers with "DB Instance Isolation"

**Schema Consolidation**
Share servers, OS and database with "Schema Isolation"

**NEW DB 12c Multitenant**
Share servers, OS and database with "Pluggable DB Isolation"
The need for Database Consolidation

Why?

- Customers have 100s or 1000s of databases across their enterprise

- They want to lower costs by operating these databases on a centrally managed platform

- With only hardware consolidation, each database has an overhead, that prevents 100s of database from being placed on the same physical server
Database Consolidation Requirements

- No change to applications
- No performance degradation
- Centralized resource management
- Isolation between environments
- Simplify patching and upgrades
Multitenant Architecture + Pluggable Databases

- Oracle Database 12c lets you have many *pluggable databases* (PDBs) in a single *multitenant container database* (CDB)
- PDBs share common resources
- The application connects to the PDB and sees it just like a pre-12c database
- The system administrator connects to the CDB as a whole and sees a single system image
Oracle Database Architecture

Each Database requires memory, processes and database files
Oracle Database Architecture

Each Databases requires memory, processes and database files
Oracle Database Architecture

Each Databases requires memory, processes and database files
New Multitenant Architecture

Memory and processes required at container level only
New Multitenant Architecture
Memory and processes required at container level only
New Multitenant Architecture
Memory and processes required at container level only
New Multitenant Architecture

Memory and processes shared and managed at container level
Managing Shared Resources

Resource management for consolidated databases

- High Priority
- Medium Priority
- Low Priority

Diagram showing resource management in a container database with priorities for different applications.
Pluggable vs Separate Databases

Highly Efficient: 6x Less H/W Resource, 5x more Scalable

OLTP benchmark comparison

Only 3GB of memory vs. 20GB memory used for 50 databases

Pluggable databases scaled to over 250 while separate database instances maxed at 50
Manage Many Databases as One

Backup databases as one, recover at pluggable database Level

One Backup

Point-in-time recovery at pluggable database level
Manage Many Databases as One

One standby database covers all pluggable databases
Simplified Patching

Apply changes once, all pluggable databases updated
Simplified Patching
Apply changes once, all pluggable databases updated
Simplified Patching

Apply changes once, all pluggable databases updated

Upgrade in-place
Simplified Patching

Apply changes once, all pluggable databases updated

Upgrade in-place
Simplified Upgrades
Flexible choice when patching & upgrading databases
Creating Databases for Test and Development

Fast, flexible copy and snapshot of pluggable databases
Creating Databases for Test and Development

Fast, flexible copy and snapshot of pluggable databases
Creating Databases for Test and Development

Fast, flexible copy and snapshot of pluggable databases
Oracle Database 12c Multitenant Architecture

Benefits

- Reduced TCO
  - Administrative costs
  - Operational costs
  - Data Center costs
  - Storage costs
  - Contingency costs

- Improved …
  - Resource utilization
  - Manageability
  - Service Management
## Oracle Database 12c
### Alternative Architectures

<table>
<thead>
<tr>
<th>Business Value</th>
<th>Many databases on one machine</th>
<th>Many databases as schemas in one database</th>
<th>Many databases as PDBs in one CDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation</td>
<td>Easy</td>
<td>Difficult*</td>
<td>Easy</td>
</tr>
<tr>
<td>Isolation</td>
<td>Highest</td>
<td>Limited</td>
<td>High</td>
</tr>
<tr>
<td>Availability</td>
<td>High</td>
<td>Highest</td>
<td>Highest</td>
</tr>
<tr>
<td>Scalability</td>
<td>Limited</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Performance</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Resource management</td>
<td>Fair</td>
<td>Severely limited</td>
<td>Excellent</td>
</tr>
<tr>
<td>ROI</td>
<td>Low</td>
<td>High</td>
<td>Highest</td>
</tr>
</tbody>
</table>

* Need to validate application schemas can co-exist
Pluggable Database

- A PDB feels and operates identically to a non-CDB Database
- You cannot tell, from the viewpoint of a connected client, if you’re using a PDB or a non-CDB Database
- A new PDB can be created in seconds
Pluggable Database

Simply unplug…

- Moving between CDBs is a simple case of moving a PDBs metadata
- PDBs can be moved using
  - SQL Developer
  - Enterprise Manager 12c
  - SQL commands
  - Database Configuration Assistant (DBCA)
Pluggable Databases
And if you can have one…

- The Pluggable Databases architecture can currently support up to 252 PDBs in a single CDB Database
- You can utilize Database Links between PDBs and between PDBs and non-CDB databases
Shared Resources

- Background processes
- Shared/process memory
- Oracle metadata
- Control files
- Redo Log files
Each PDB has its own set of tablespaces including SYSTEM and SYSAUX

PDBs share UNDO, REDO and control files

By default the CDB has a single TEMP tablesapce, but PDBs may create their own
Users

- PDB Users are the successors for users in a non-CDB
- Local users can administer PDBs
- A special class of users “Common Users” has the same identity in all PDBs
- Common users can log into any database they have “create session” privilege for
- Common users own the Oracle system
- PDBs can be administered by a common user, too
Application Express
in
Oracle Database 12c Multitenant Architecture
Standard “Default” 12c Installation
Application Express 4.2 installed in CDB (Container DB)

- APEX 4.2 is installed into CDB$Root and PDB$Seed
- Each PDB contains meta-data linked objects to APEX Schema in Root
- Each PDB also has an APEX Schema which holds local tables

Local Tables (APEX meta-data for *that* PDB)

Meta-data Links

Views, Packages, Functions, Procedures
Standard 12c Installation
Creating a new PDB from seed

- PDB created by copying PDB$Seed
- New PDB includes Local Tables and Meta-data link to Root
- Web Listener *must* be configured for each PDB {each with unique port}
Removing Common Application Express

Uninstalling Application Express 4.2 from the CDB

1) Post-installation run `apxremov_con.sql`* to remove from CDB, PDB Seed and all PDB’s {*DB Patch 16946990 required}

*All existing APEX meta-data within all of the PDBs will be removed*
Non-Standard Configuration
Installing Different Versions into PDBs

1) Uninstall Application Express from the CDB
2) Log into PDB, run `apxinstall.sql` or `apxrtins.sql`

*Can install different versions of APEX (> Release 4.2) into any PDB*
Re-Installing into Root Database
Application Express 4.2 *removed* and then reinstalled into CDB

1) Uninstall Application Express from the CDB
2) Run script `apexins_con.sql` or `apxrtins_con.sql`

*Use to install runtime only* Application Express

*Installs in Root, PDB$Seed and all PDBs, also creates meta-data links*
Migrating from Oracle Database 11g to PDB

1) Upgrade 11g database to 12.1 in place
2) Place the non-CDB into read only mode
3) Connect to non-CDB and generate an XML Metadata file
4) Shutdown the non-CDB
5) Plug in non-CDB to CDB
6) Run `noncdb_to_pdb.sql` → Replaces local APEX_040200 schema objects with meta-data linked objects {If APEX configured in CDB$Root}
Moving/Copying PDBs Between 12c Databases
From APEX IN CDB$Root → APEX IN CDB$Root

1) Move / copy PDB
2) Configure Web Listener for new PDB \{Must specify a unique port\}
Moving/Copying PDBs Between 12c Databases

From APEX **NOT IN** CDB$Root → APEX **IN** CDB$Root

1) Move / copy PDB

*Application Express installed differently in source and target databases*
Moving/Copying PDBs Between 12c Databases
From APEX NOT IN CDB$Root → APEX IN CDB$Root

2) Run `apex_to_common.sql` in PDB to create meta-data links
3) Configure Web Listener for new PDB
Moving/Copying PDBs Between 12c Databases

No APEX → APEX **IN** CDB$Root

1) Run `apexins.sql` on PDB while in *Source* database

*Need to install **before** moving as can’t be installed into PDB when in Root*
Moving/Copying PDBs Between 12c Databases

No APEX → APEX \textbf{IN} CDB$Root

2) Move / copy PDB

\textit{Now that Application Express is in source database it is ready to move}
Moving/Copying PDBs Between 12c Databases
No APEX → APEX IN CDB$Root

3) Run `apex_to_common.sql` in PDB to create meta-data links
4) Configure Web Listener for new PDB
Moving/Copying PDBs Between 12c Databases

From APEX **NOT IN** CDB$Root → APEX **NOT IN** CDB$Root

1) Move / copy PDB
2) Configure Web Listener for new PDB
Moving/Copying PDBs Between 12c Databases
From APEX **IN** CDB$Root → APEX **NOT IN** CDB$Root

- Contact Oracle Support
- Special version specific `apex_to_local.sql` scripts available
Moving/Copying PDBs Between 12c Databases

**Incompatible Versions**: Higher Release $\rightarrow$ Lower Release ($x > y$)

1) Run `apxpatch_con.sql` or `apexins_con.sql` in target CDB to upgrade CDB Root to be the same as the source

*Need to ensure that both target and source databases are the same release*
Moving/Copying PDBs Between 12c Databases

Incompatible Versions: Higher Release $\rightarrow$ Lower Release ($x > y$)

2) Move / copy PDB

Now they are the same release, can move the PDB and perform any additional steps, such as configuring the Web Listener
**Moving/Copying PDBs Between 12c Databases**

**Incompatible Versions:** Lower Release $\rightarrow$ Higher Release ($x < y$)

1) **Move / copy PDB**

*These steps are only if the source database can’t be updated*

*If possible, update the source and then perform a straight move / copy*
Moving/Copying PDBs Between 12c Databases

**Incompatible Versions:** Lower Release → Higher Release \((x < y)\)

2) Run `catcon.pl` in new PDB to upgrade to the same release as in the target CDB Root installation

See Application Express 4.2 in Oracle DB 12c Installation Guide for syntax
Upgrading / Patching Application Express
APEX **IN** CDB$Root versus APEX **NOT IN** CDB$Root

- Run container scripts, such as `apexins_con.sql`, in CDB
  
  Perform update / patch **ONCE** for CDB and all PDBs

- Run standard scripts, such as `apexins.sql`, in PDBs
  
  Perform update / patch on **EACH** PDB separately & independently
Oracle Database 12c
New Features
Important to
Application Express
Developers
New Feature - Varchar2 32K

- 12c Database Parameter – max_string_size
  - STANDARD [Default] = 4000 byte/character limit (Pre-12c behavior)
  - EXTENDED = 32767 byte/character limit

- After changing parameter must bounce database

- Can specify textfield or textarea sizes up to 32K (32767)

- Can run /core/collection_member_resize.sql to change collection VARCHAR2 columns from 4000 characters to 32767 characters

```
ALTER SYSTEM set MAX_STRING_SIZE = EXTENDED scope = SPFILE
```
New Feature - Default | Default On Null

- Default values are utilized when inserting a record if column not specified in the insert statement
- With ON NULL will use default even if specified in the insert statement
- If you specify a value explicitly it will not use default
- Avoids the need for trigger logic

```
CREATE TABLE myemp (employee_id number, last_name varchar2(25),
                     department_id NUMBER DEFAULT ON NULL 50 NOT NULL);
```
New Feature - Identity Column

- Use Identity Columns instead of specifying a Sequence and using triggers / processes to retrieve the next value from the sequence
  - ALWAYS [Default] – Uses the sequence generator
  - BY DEFAULT – Can explicitly assign a specified value
  - BY DEFAULT ON NULL – Can explicitly assign, but uses sequence if evaluates to NULL
- SQL Workshop → Can specify Identity in Create Table Wizard
- Create Form / Report Wizard → Creates correct item type {display only}

```
CREATE TABLE t1 (c1 NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY,
c2 VARCHAR2(10));
```
New Feature - Invisible Column

- Invisible columns are *user specified* hidden columns
  - SELECT * → Won’t display invisible columns
  - INSERT INTO x VALUES → Won’t insert values in invisible columns
  - *Must explicitly specify invisible columns to include them*

- SQL Workshop → Not shown in Object Browser, Query Builder, etc.

- Create Form / Report Wizards → Columns won’t be shown

- Must *manually* add to Source within APEX Region (post-generation) to view in Reports, etc.
New Feature - Limiting Rows from SQL Query

- FETCH returns top ‘N’ records
- Can specify rows or percentage,
- Can include OFFSET to skip specified number of rows
- Respects ORDER BY clause

```sql
SELECT employee_id, last_name
FROM employees
ORDER BY employee_id
FETCH FIRST 5 ROWS ONLY;
```

```sql
SELECT employee_id, last_name
FROM employees
ORDER BY employee_id
OFFSET 10 ROWS FETCH NEXT 5 ROWS ONLY;
```
New Feature - Data Redaction
Mask Application Data Dynamically

<table>
<thead>
<tr>
<th>Policy enforced redaction of sensitive data</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Soc. Sec. #</th>
<th>DOB</th>
<th>PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>115-69-3428</td>
<td>11/06/71</td>
<td>5623</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE NAME</th>
<th>SOC. SEC.#</th>
<th>DOB</th>
<th>PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SARA JONES</td>
<td>115-69-3428</td>
<td>11/06/71</td>
<td>5623</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE NAME</th>
<th>SOC. SEC.#</th>
<th>DOB</th>
<th>PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAMES SMITH</td>
<td>114-50-2356</td>
<td>01/12/69</td>
<td>7703</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLIENT CONTACT</th>
<th>SOC. SEC. NO.</th>
<th>DOB</th>
<th>PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SARA JONES</td>
<td>*<strong>-</strong>-3428</td>
<td>01/01/01</td>
<td>****</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLIENT CONTACT</th>
<th>SOC. SEC. NO.</th>
<th>DOB</th>
<th>PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAMES SMITH</td>
<td>*<strong>-</strong>-2356</td>
<td>01/01/01</td>
<td>****</td>
</tr>
</tbody>
</table>

Payroll Processing
Call Center Operator
Proof Point - ARIA People

1st “Production” application on Oracle Database 12c

- Written in Application Express
- Used by virtually every employee in Oracle
- >1.3 million page views / day
- > 60 page views / sec
- Median execution time 0.05 seconds