TECHNICAL WHITE PAPER

# Oracle Enterprise Platform Modernization

Notes from a staged Oracle upgrade: database, then operating system, then E-Business Suite.

**Database · Operating System · Oracle E-Business Suite**

DBA and Enterprise Applications Practice · QCS Tech Solutions Inc.

**12.1 → 19c**
Oracle Database, non-CDB to CDB/PDB

**OL7 → OL8**
Oracle Linux operating system

**R12.1.3 → R12.2.14**
Oracle E-Business Suite

**3 stages**
each with entry checks, sign-off and evidence

DATABASE · OPERATING SYSTEM · E-BUSINESS SUITE

# Contents

**Database first, OS next, EBS last. Each stage got its own entry checks, its own sign-off and its own evidence.**

This paper walks through an Oracle upgrade on a long-established environment and points out where things tend to go wrong, using what actually happened on the project. It is not a runbook: take commands, patch steps and supported upgrade paths from Oracle's documentation for your exact release.


| | | |
|---|---|---|
| – | Abstract, scope and approach | 03 |
| 01 | Database upgrade: Oracle 12.1.0.2 to 19c | 04 |
| 02 | Operating-system upgrade: Oracle Linux 7 to Oracle Linux 8 | 07 |
| 03 | Oracle E-Business Suite upgrade: R12.1.3 to R12.2.14 | 09 |
| 04 | Problems faced during the database and EBS upgrade | 12 |
| 05 | Conclusion | 14 |
| 06 | References and further reading | 14 |

ABSTRACT · SCOPE AND APPROACH

# Three layers, moved in a set order

This paper walks through an Oracle upgrade we carried out on a fairly old, well-established environment. Three layers had to move: the database, the operating system and E-Business Suite.

| WORKSTREAM | SOURCE ENVIRONMENT | TARGET ENVIRONMENT |
|---|---|---|
| **Database** | Oracle Database 12.1.0.2, non-CDB | **Oracle Database 19c, CDB/PDB** |
| **Operating system** | Oracle Linux 7 | **Oracle Linux 8** |
| **Oracle E-Business Suite** | EBS R12.1.3 | **EBS R12.2.14** |

We did it in a set order. Database first, OS next, EBS last. There's a reason for that: EBS leans on the database's service setup and on a certified OS and middleware stack, so if either of those is shaky, the application upgrade inherits the problem. Every stage got its own entry checks and its own sign-off, and we kept evidence for each one.

## Implementation sequence

STAGE 1

### Database upgrade

`12.1.0.2 → 19c`

STAGE 2

### Operating-system upgrade

`Oracle Linux 7 → 8`

STAGE 3

### EBS upgrade

`R12.1.3 → R12.2.14`

The database stage gives you the target version and the PDB service mapping. The OS stage then proves the servers, libraries, filesystems and runtimes are fit for the new Oracle stack. EBS only starts once both are checked off.

> **Approach.** Don't read this as a runbook. There are no step-by-step commands here. What we've tried to do is point out where things tend to go wrong, using what actually happened on the project. Commands, patch steps and supported upgrade paths change between releases, so take those from Oracle's documentation for whatever release and patch level you're on.

STAGE 1 · DATABASE

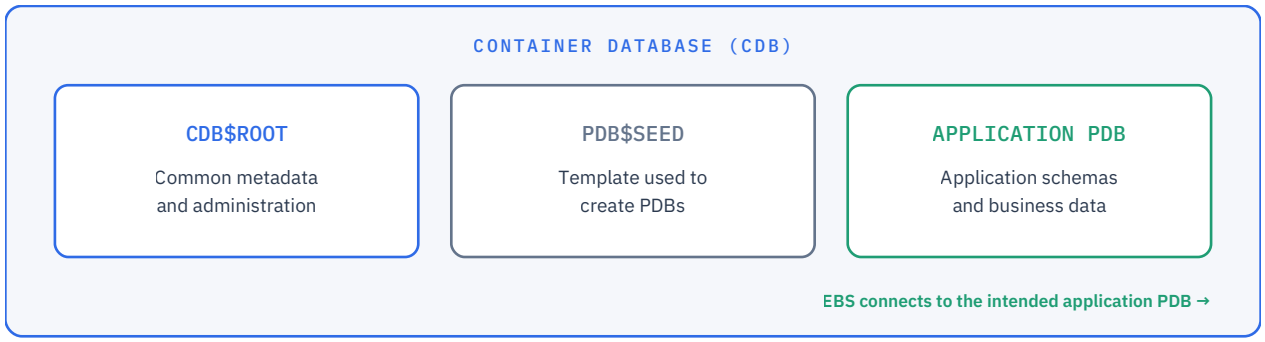
# Database upgrade: Oracle 12.1.0.2 to 19c

Everything else in the programme sat on top of the database, so it went first.

We were moving from 12.1.0.2 as a non-CDB to 19c on multitenant. Installing new binaries is honestly the small part of that. The bigger shift is that application data now lives inside a PDB, and that knocks on to service registration, admin routines, backup and recovery, monitoring and how the application connects in.

FIGURE 1

## Oracle Database 19c multitenant architecture




Application data is held in pluggable databases within a container database: the CDB and the application PDB.

### 1.1 Understanding the CDB/PDB model

Quick refresher. A CDB has a root container, a seed PDB and one or more application PDBs. Common metadata and admin functions sit in the root. The seed is just a template for spinning up new PDBs. Your application schemas and business data go in the application PDB. For EBS, the bit that really matters day to day is making sure connections end up in the right PDB.

It's tempting to leave services until the end and treat them as a listener tweak. We'd advise against that. Check the service name, which PDB it maps to and whether it's registered with the listener properly, and then test it from the app server with the exact connect string the application is going to use.

### 1.2 Source assessment and target preparation

Take a proper baseline of the source database before you start. That means the component registry, character set, timezone-file version, invalid objects, tablespace usage, options and the recovery setup. Also note the init parameters, memory, process and session limits, filesystem layout and Oracle home details. Half the value of this is afterwards, when something looks odd and you need to know whether it was already like that.

Build the target Oracle home and double-check certification against both the application and the OS version you're heading to. Size for datafiles, temp, the recovery area, archive logs and growth. We've seen upgrades that finish without a single error and then run out of space or memory the first week they carry real production traffic.

### 1.3 Upgrade planning

01 Confirm the upgrade path is supported and that you've met every pre-upgrade requirement for your exact versions.

02 Run the pre-upgrade checks early. Fix what they flag, or at least write it down, before the window opens.

03 Rehearse on a clone that looks like production, CDB/PDB conversion and all, if you possibly can.

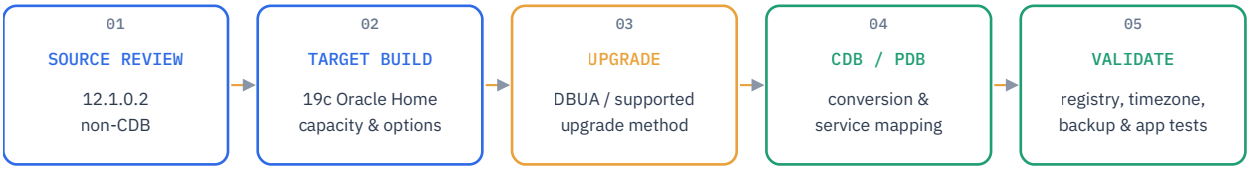
04 Agree on backup, recovery and rollback before go-time. Hang on to every log and report the upgrade produces.

### 1.4 Database upgrade execution and validation

We didn't call the database stage done until the database, its services and the app's connection path had all been tested end to end.

FIGURE 2

## Database upgrade: a controlled workflow




From assessing the source through to post-upgrade checks.

### Timezone migration

Give the timezone-file migration its own check. Ours didn't complete through DBUA the way it was supposed to, and someone had to go in and finish it manually. So check the upgrade state before and after. If a window's been left open or half-done, close it out with the supported procedure. Then actually look at the final timezone-file version. Don't assume it's fine just because the main upgrade said it succeeded.

### Post-upgrade checks

| AREA | WHAT TO VERIFY |
|---|---|
| **Database state** | CDB and PDB open modes, plug-in compatibility, registry components and alert-log errors |
| **Objects** | Invalid objects after recompilation; separate Oracle-owned objects from custom and third-party schemas |
| **Services** | Listener registration, service-to-PDB mapping and connection tests from application hosts |
| **Capacity** | Tablespace headroom, FRA usage, archive-log generation, memory and process/session capacity |

| AREA | WHAT TO VERIFY |
|---|---|
| **Recovery** | Backup completion, recovery configuration and restore readiness; flashback settings where configured |
| **Application** | Connections through the actual EBS database service and representative database-dependent operations |

### What the database sign-off should establish

By sign-off, the database should be stable, the PDBs you need should be opening in the right mode, and the application services should be hitting the correct PDB. Tablespace and recovery-area space should comfortably cover the expected load. Somebody should have looked over the backup and recovery settings, and the app team should have connected through the normal service names and said yes, it works.

Keep all of it with the upgrade record: pre-upgrade reports, DBUA or command-line logs, registry queries, PDB status, service checks, the invalid-object review, backup results and the connection tests. It's a bit of admin, but it saved us more than once later on. When something broke during the OS or EBS stage, we could see exactly what the database looked like when that stage began.

> COMMON AREAS THAT NEED ATTENTION
>
> The upgrade utility is rarely what holds things up. It's the stuff around it.

Timezone state, invalid objects, running short on tablespace or FRA, service registration, and the little ways test and production never quite match. Chase each finding and note how it ended: fixed, accepted as an exception, or parked with a named owner.

STAGE 2 · OPERATING SYSTEM

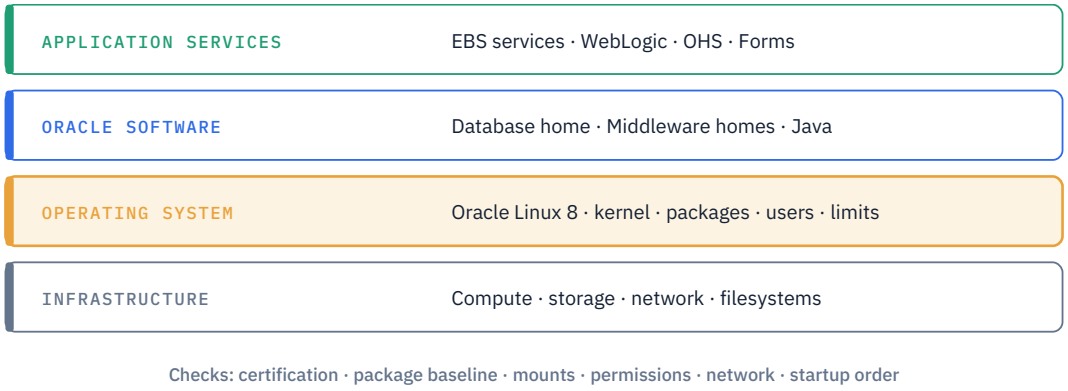
# Operating-system upgrade: Oracle Linux 7 to Oracle Linux 8

The OS went second, once the target database and the connection model were nailed down.

Stage two took the servers from Oracle Linux 7 to 8. It looks like an OS install on paper. In practice it's mostly about compatibility and configuration, because Oracle homes, middleware, Java, system libraries, service definitions, mounts, security settings and networking can all shift underneath you.

FIGURE 3

The OS layer and the Oracle stack that depends on it




> Validate the operating-system layer and the Oracle stack above it.

## 2.1 Readiness before the OS change

Note down what you've got now: release and kernel, packages, users and groups, limits, mount points, network settings and whatever starts at boot. Same for the Oracle side, so ownership and permissions on the homes, inventory pointers, environment profiles, symlinks and service scripts. Nobody pays attention to this stuff when the conversation is all about the OS version. Then something behaves differently after the change and this list is where the answer usually is.

Check certification for the actual versions you run, database, EBS, middleware and any third-party bits. Look at Java and runtime requirements and package or library dependencies too. A vendor saying they support Oracle Linux 8 doesn't count for much if your patch level or runtime isn't the certified combination.

## 2.2 Infrastructure and configuration checks

C1 STORAGE

Filesystem space, how storage is presented, mount options, and who owns the Oracle and application directories.

C2 KERNEL & SERVICES

Kernel parameters, user limits, required packages, systemd service definitions and what depends on what at startup.

C3 NETWORK

DNS and hosts resolution, listener access, firewall rules, routes and time sync.

C4 MULTI-NODE

Shared filesystems and node-specific mounts should behave identically on every database and app node.

## 2.3 Validation after the OS change

Once the new OS is in, start the database and application components in the planned order. Watch the logs for missing libraries, permission changes, mount problems or defaults that quietly changed. Get the Oracle homes and app-tier services up cleanly, then test the database connection from the app tier. The server booting is nice, but it doesn't tell you much about whether Oracle is ready.

Push some normal EBS transactions through and test the interfaces the business actually uses. On a multi-node system, check every node against the same baseline. It only takes one node with a missing package, an odd mount or a wrong permission to give you intermittent faults that nobody can reproduce, the kind that turn up under load or in a failover and nowhere else.

> **OS stage completion.** We signed off the OS stage when the release and kernel were confirmed, packages and limits were in place, filesystems and permissions checked out, Oracle came up cleanly and the database and app tiers could reach each other. Keep the before/after OS inventory and the service test results. You'll want them in the EBS upgrade record.

STAGE 3 · E-BUSINESS SUITE

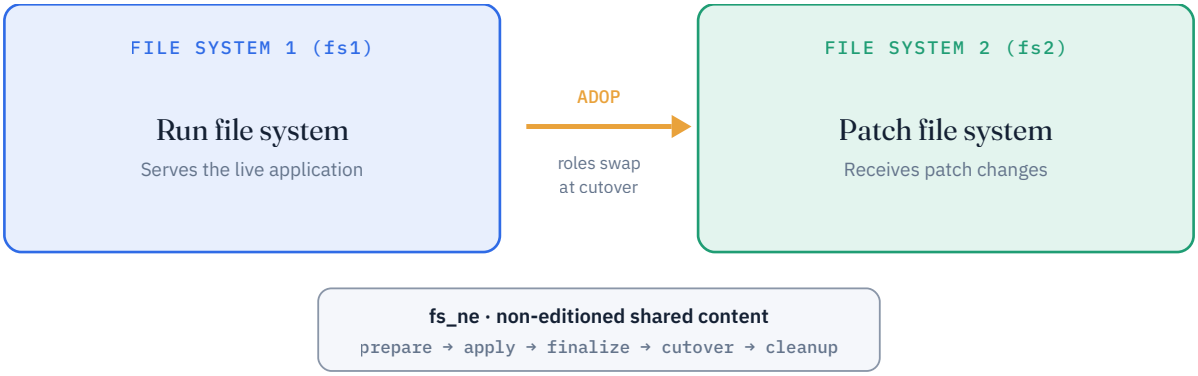
# Oracle E-Business Suite upgrade: R12.1.3 to R12.2.14

EBS came third, after the database and OS were done.

The EBS upgrade wasn't just new application code. We had to deal with database objects, middleware, web access, Forms, OACore, concurrent processing, Workflow and a handful of external integrations. R12.2 also brings online patching with it, which means the dual file system and the ADOP cycle are now part of normal maintenance whether you like it or not.

FIGURE 4

## The EBS R12.2 dual file system




ADOP uses alternating run and patch file systems; fs_ne holds non-editioned content. Run and patch swap roles at cutover.

### 3.1 R12.2 dual file system and ADOP

R12.2 runs on two file systems, fs1 and fs2, plus fs_ne for shared content that isn't editioned. At any moment one of them is serving users and the other is the patch side. ADOP drives the cycle (prepare, apply, finalize, cutover, cleanup), and the roles flip at cutover. Sounds obvious, but before you start a patch cycle or try to debug a broken one, make sure everyone knows which file system is which right now.

Compare the context values on fs1 and fs2, particularly anything to do with host, domain and ports. A mismatch there can upset AutoConfig, web routing and the sync between file systems. That's exactly what we ran into. Some domain values had drifted, and running an empty ADOP cycle brought the two sides back together. It worked for us. We wouldn't treat it as a general fix.

### 3.2 Patch preparation and execution

01 AD and TXK code levels, RUP prerequisites, what's in the patch, and the state of the last ADOP cycle.

02 App-tier services, node configuration, shared filesystem access and Node Manager, all checked before fs_clone.

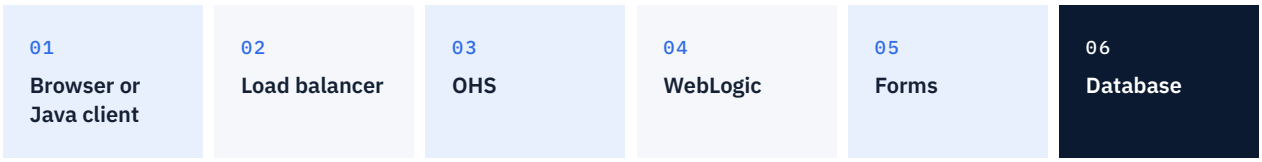
03 Track progress with ADOP status and the worker logs. If a worker fails, save the whole error stack.

04 Run AutoConfig after any supported context change and check what it generates for OHS and the app tier.

05 Don’t touch ADOP worker or session state by hand unless Oracle Support has agreed a recovery procedure with you.

### 3.3 Web tier and Forms access

Test the route a real user takes, from the browser or Java client through the load balancer, OHS, WebLogic and Forms and down to the database. Getting a login page only means the login path works. Forms can still fall over on servlet routing, managed-server endpoints, SSL termination, DNS or the way Java launches on the user’s laptop.




### 3.4 Application-tier validation and implementation lessons

**Forms, OACore and concurrent processing.** For a Forms launch failure or a frmservlet 404, go hop by hop. Web-entry host and domain, protocol and port, the OHS location, the WebLogic target, whether the Forms managed server is actually running, the SSL wallet and cert chain, load-balancer health, DNS, and what the client does with the JNLP file. One thing we learned: if Forms works over VNC but not from a laptop, compare those two paths first. Don’t start changing server settings until you have.

Look at OACore and concurrent-processing JVM settings alongside heap usage, GC behaviour, the app logs and how much memory the host has. We saw clear heap pressure and moved the OACore heap to a 2 GB minimum and 4 GB maximum. That was based on what the workload was doing and what the box could spare. Bumping the heap isn’t a cure for every Java error, and it shouldn’t be the first thing you reach for.

**Patch data and recovery.** The R12.2.14 RUP threw two data problems at us. One was a Trade Management script, `OZFUBDG.sql`. The other was duplicate rows in an ICX table. What we did to fix them was specific to those cases, so please don’t lift it as a standard fix. Work out which SQL is failing and what data it’s touching, take a backup, check what Oracle Support says and test before you apply anything.

There was also a Workflow migration failure involving `WFPWDMIG.sql` and `ECX_OBFUSCATE.DECRYPT_AES256`, and that one went to Oracle Support. If a failure touches seeded encryption routines or leaves ADOP workers stuck halfway, keep the session details and worker logs and stick to the supported recovery path. Editing patch tables by hand is asking for trouble.

### EBS acceptance checks

| AREA | EVIDENCE TO RETAIN |
|---|---|
| **Patching** | ADOP cycle status, worker logs, AD/TXK levels and RUP inventory |
| **Configuration** | AutoConfig results, context consistency, OHS and WebLogic configuration |

| AREA | EVIDENCE TO RETAIN |
| --- | --- |
| **Core services** | Forms, OACore, concurrent processing and Workflow status |
| **Business access** | Login, Forms launch, responsibilities and representative business transactions |
| **Integrations** | EDI, APIs, 3PL, WMS, OPM, UKG, Concur and Demantra checks as applicable |

FIELD NOTES · ISSUES LOG

# Problems faced during the database and EBS upgrade

Not everything went to plan. Some issues needed a lot more digging than the standard upgrade steps cover.

These all happened on the actual project, and where we mention a fix, it's one we really applied.

## Database upgrade and post-upgrade issues

### Timezone migration did not finish through DBUA

RESOLVED

`ORA-56920`

The database was stuck in an active DST upgrade state (we got ORA-56920 when we tried another prepare), so we had to work out where the timezone upgrade had got to and finish it by hand before moving on.

### PDB plug-in from XML descriptor

`ORA-65139`

Plugging a PDB in from its XML descriptor during the CDB/PDB work came back with ORA-65139. The descriptor and compatibility details needed more investigation. Get PDB plug-in errors sorted before you point any application services at the new PDB.

### Statistics gathering failed on locked tables

RESOLVED

`ORA-20005`

Stats gathering failed because statistics were locked on a bunch of tables. We had to find the locked objects and handle statistics as its own task, separate from recompiling and validating the upgrade.

## Oracle E-Business Suite upgrade and application-tier issues

### Trade Management script failed during the RUP

RESOLVED

`OZFUBDG.sql`

OZFUBDG.sql failed on a Trade Management column during the R12.2.14 RUP. We renamed that column to get past the specific clash with the script. Read seeded patch scripts carefully before you change anything in the data dictionary.

### Duplicate rows in an ICX table

RESOLVED

The ICX table error turned out to be duplicate rows. Once we'd found the conflicting data, we deleted one duplicate row and re-ran the patch step.

### Workflow migration worker failure

WITH ORACLE SUPPORT

WFPWDMIG.sql · ORA-20006 · ORA-20001 · ORA-28817 (ECX_OBFUSCATE.DECRYPT_AES256)

The Workflow migration worker died in WFPWDMIG.sql. We logged it with Oracle Support. Until Support comes back with a final answer, it shouldn't be described as resolved.

### Context values drifted between fs1 and fs2

RESOLVED

Some domain-related context values didn't match between fs1 and fs2, and that threw off the app-tier configuration. An empty ADOP patch cycle got the file systems back in line.

### Forms returned frmservlet 404 while login worked

We went through OHS routing, the Forms managed-server endpoint, the web-entry host and domain, SSL and wallet setup, name resolution and load-balancer access. Oddly, Forms over Java/JNLP behaved differently on a laptop compared with a VNC session.

### Node Manager port in use; Java heap-space errors

PORT 5557 · OACORE HEAP 2 GB / 4 GB

The multi-node fs_clone validation hit Node Manager port 5557 already being in use. Separately, OACore and concurrent-processing logs were showing Java heap-space errors. In that environment we raised the OACore heap to 2 GB initial and 4 GB max.

CONCLUSION · WHAT WE'D PASS ON

# Test the joins between the layers

Database, then OS, then EBS. Keeping to that order meant each stage had solid ground under it.

If there's one thing we'd pass on, it's to test the joins between the layers just as hard as the layers themselves.

THE JOINS THAT MATTER

### Database ↔ services

The services need to point at the right PDB.

### OS ↔ Oracle stack

The OS has to support the Oracle stack.

### EBS ↔ users

EBS has to work through the same web and app paths your users go through every day.

REFERENCES AND FURTHER READING

1. Oracle Database 19c Upgrade Guide, Part No. E96252-83
2. Oracle Database 19c Database Concepts, Part No. E96138-11
3. Oracle E-Business Suite Upgrade Guide, Release 12.0/12.1 to 12.2, Part No. E73540-26
4. Oracle E-Business Suite Maintenance Guide, Release 12.2, Part No. E22954-33
5. Oracle E-Business Suite Online Patching documentation (ADOP)
6. Oracle Linux 8 Documentation

A note on the references: those are Oracle publication part numbers, not My Oracle Support (MOS) Doc IDs. Use documentation and support notes that match your exact releases and patch levels. The project-specific fixes in this paper are just that. They aren't general instructions for changing Oracle application data.

ABOUT QCS TECH SOLUTIONS INC.

# Change that is controlled, tested at every layer and properly documented.

QCS Tech Solutions Inc. helps organisations modernise and look after their enterprise platforms. On the Oracle side that covers database upgrades, platform and OS moves, E-Business Suite lifecycle work, performance and availability, backup and recovery, security and ongoing support, so the system can still be supported well after the project team has gone.

ABOUT THE AUTHORS

## DBA and enterprise applications practice

The authors support Oracle databases and E-Business Suite environments day to day.

EXPERIENCE

## Upgrades and multitenant

Version upgrades, multitenant setups and Linux platform changes.

EXPERIENCE

## Online patching and app tier

Online patching, app-tier operations and troubleshooting.

THIS PAPER

## Grew out of real work

The decisions, problems and checks from modernising one integrated Oracle environment.

To discuss an Oracle database, platform or E-Business Suite upgrade, contact the QCS team.

QCS · TECH