VISCOSITY

NORTH AMERICA

Thank you for registering for a
Viscosity hosted webinar!

Viscosity Master Class

Up Next: Tuesday, November 17, 2020 | 11:00 AM - 12:00 PM Central

REGISTER NOW

https://viscosityna.com/viscosity-master-class/

Oracle Database Expert Panel

Tuesday November 17, 2020 | 11:00 AM CT – 12:00 PM CT

Charles Kim

Founder and CEO, Viscosity North America

Oracle ACE Director

Rich Niemiec

Chief Innovation Officer, Viscosity North America

Oracle ACE Director

Maria Colgan

Distinguished Product Manager, Oracle Corporation

Andy Rivenes

Product Manager Database In-Memory, Oracle Corporation

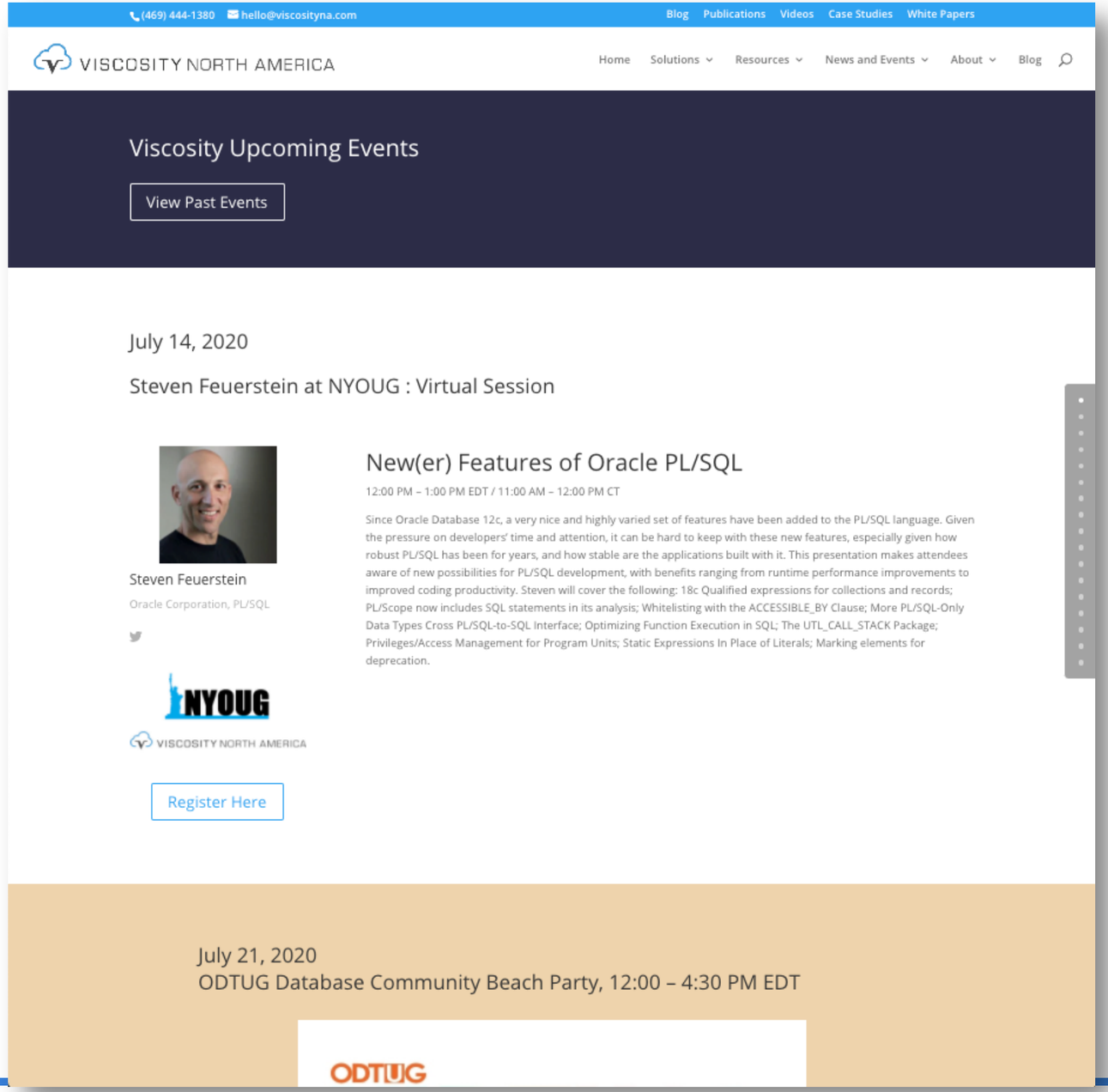
Troy Ligon

Oracle ACE & Enterprise Architect, President of SOUG

Stay up-to-date with our upcoming webinars:

Visit
viscosityna.com/event

Follow us on
LinkedIn, Facebook or
Twitter: **@viscosityna**



Running Oracle Upgrades on Docker

Sean Scott

25 years working with Oracle technology

UTOUG Board ⁘ RAC SIG Board

Oracle OpenWorld ⁘ Collaborate/IOUG ⁘ Regional UG

RAC/MAA ⁘ DR/HA ⁘ TFA/AHF ⁘ Exadata/ODA

Upgrades ⁘ Migration ⁘ Cloud ⁘ Automation

DevOps ⁘ Infrastructure as Code

Containers ⁘ Virtualization

ORACLE Certified Professional

ORACLE Certified Specialist

ORACLE
ACE Program

500+ technical experts helping peers globally

The **Oracle ACE Program** recognizes and rewards community members for their technical contributions in the Oracle community

3 membership tiers

ORACLE ACE Director | ORACLE ACE | ORACLE ACE Associate

Oracle Groundbreakers

Nominate
yourself or someone you know:
acenomination.oracle.com

For more details on Oracle ACE Program:
bit.ly/OracleACEProgram

Connect: **oracle-ace_ww@oracle.com** Facebook.com/oracleaces @oracleace

Concepts

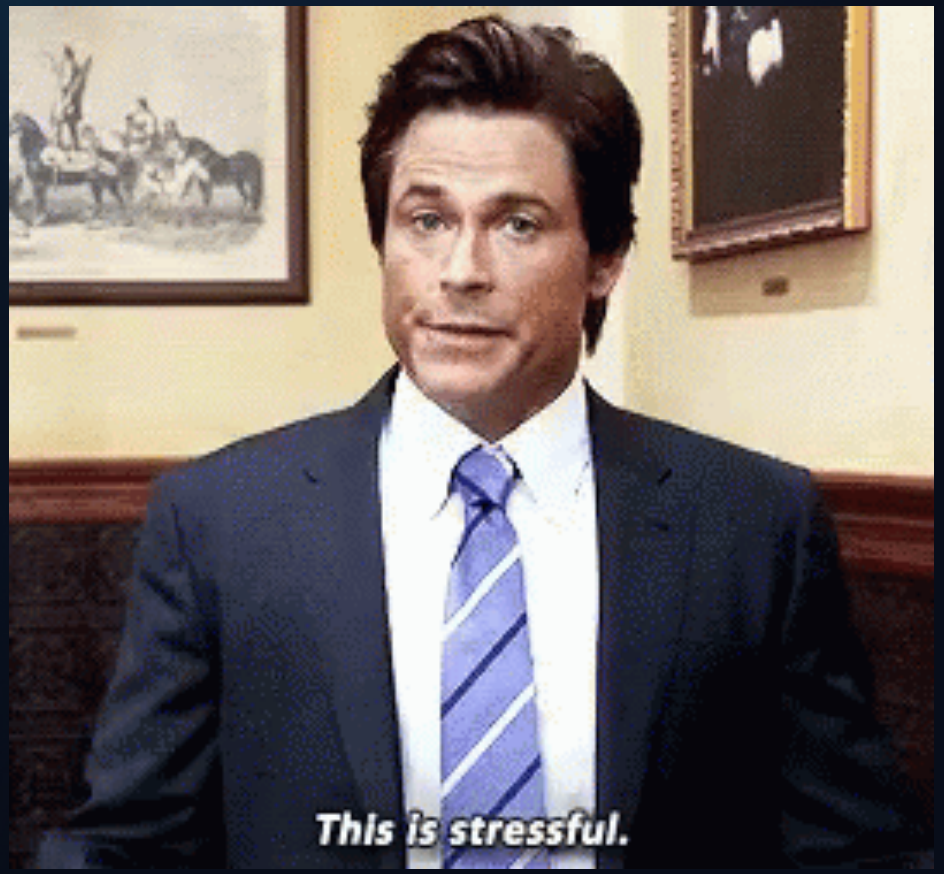
- Image
- Container
- Dockerfile - FROM, COPY, RUN, ENV
- `docker build`
- `docker run`
- `docker exec, docker attach`
- github.com/oracle/docker-images

ORACLE

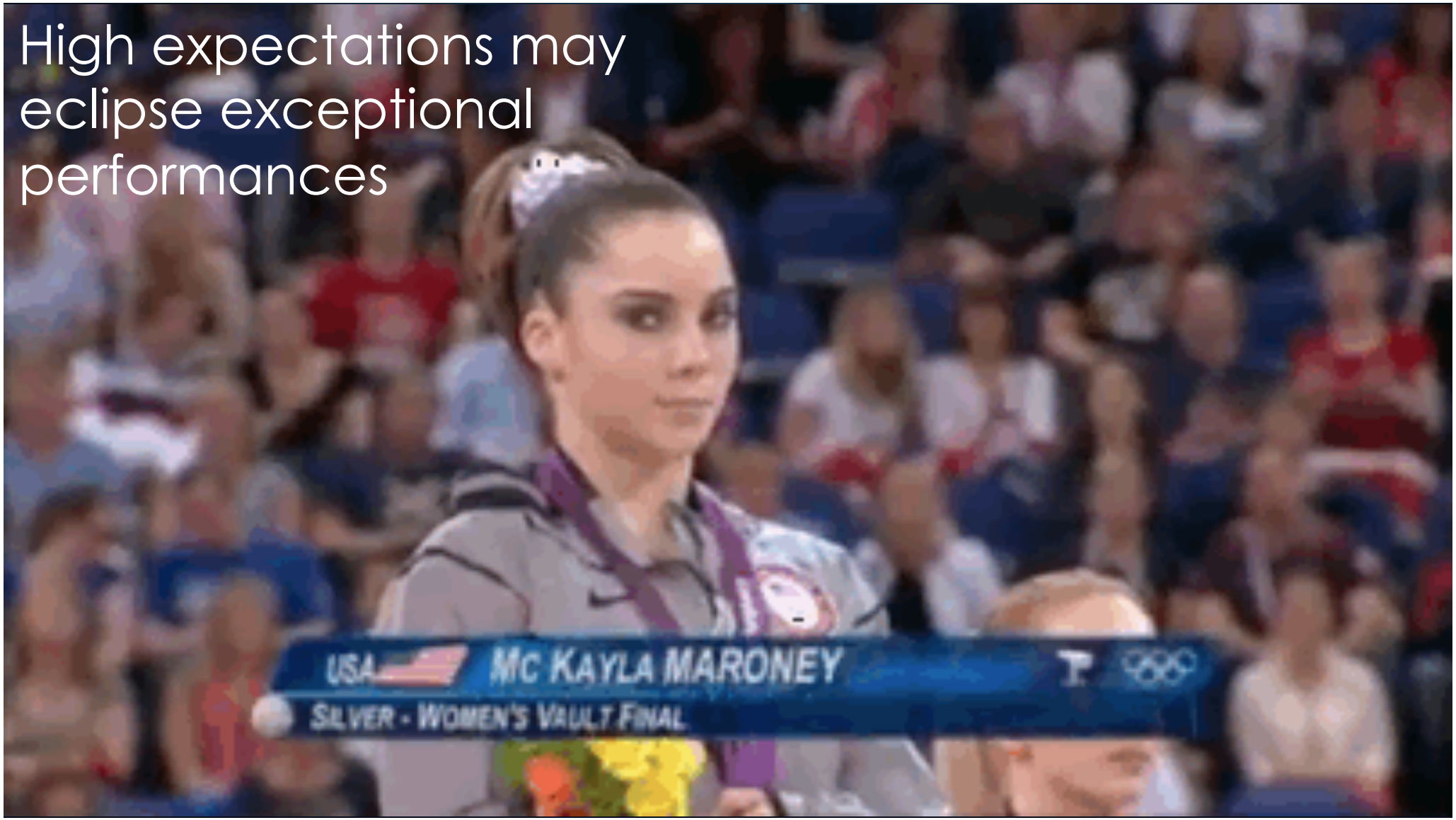
Database Upgrades:
The World Cup for DBAs

World Cup/Olympics vs. Upgrades

- High stakes
- Infrequent
- Attention
- Scrutiny
- Pressure!



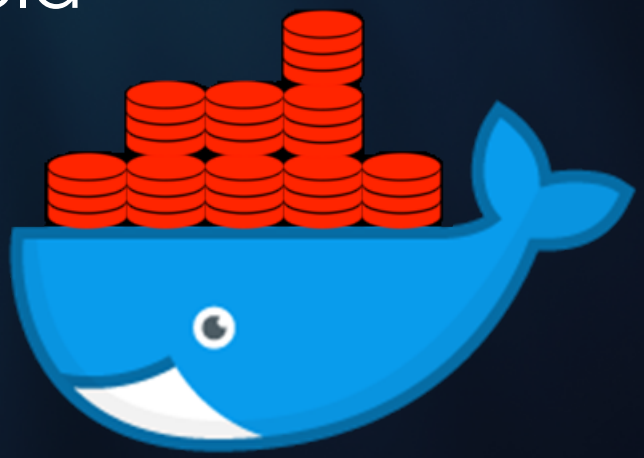
Everyone's suddenly an expert!


High expectations may eclipse exceptional performances
USA
Mc Kayla Maroney
Silver - Women's Vault Final

Train for Upgrades

Docker: The Practice Field

- Free
- Fast
- Easy

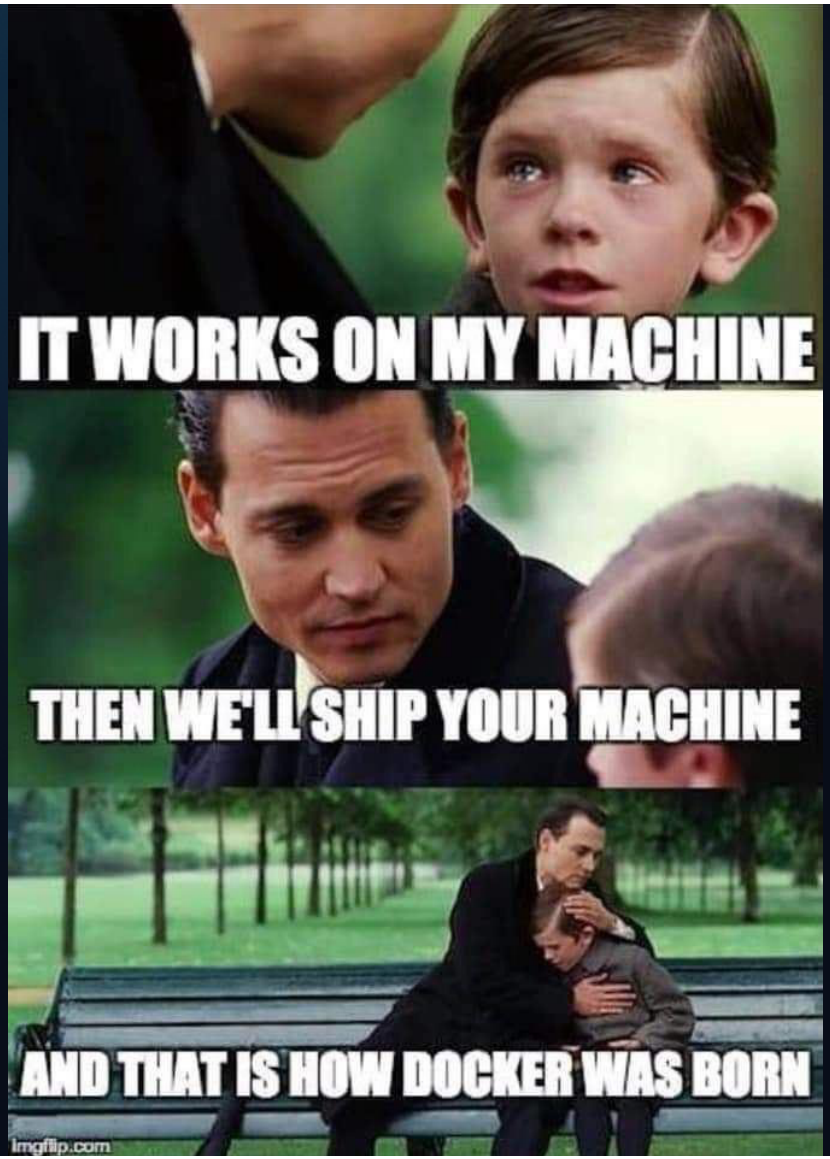


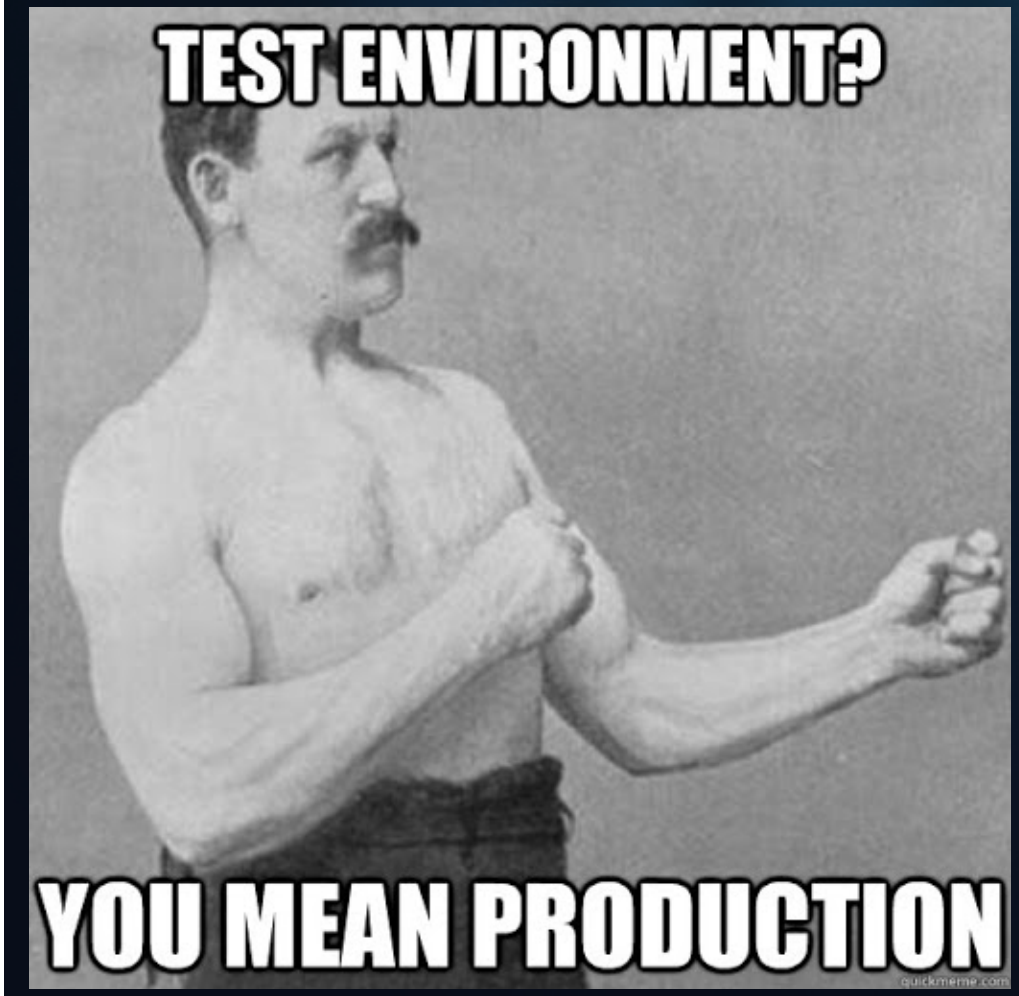
docker

ORACLE

Docker: The Practice Field

- Portable
- Versionable
- Distributable
- Configurable
- Reproducible
- Convenient





We Don't Test in Production or Deploy on Fridays (and Other Lies)

Internal Customers are Still Customers

- Dev is production for development
- Test is production for testing
- QA is production for QA

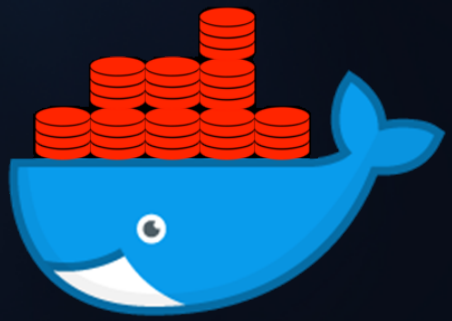
Pre-production environments support pre-production efforts that end up in… production.

If you can't take and break an environment for an unspecified, extended period, without consequences, it's not really test.

Docker - Scope

Docker Helps Build Processes

- Choosing a method
- Writing procedures and documentation
- Creating automation
- Adjusting monitoring



docker

ORACLE

Docker - Scope

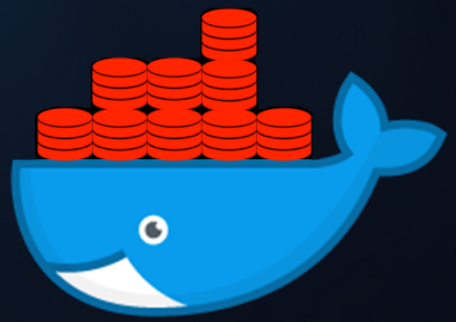
Docker is a Window Into Functional Changes

- Preview new features
- Understand changes
- Recognize new behaviors, patterns

docker

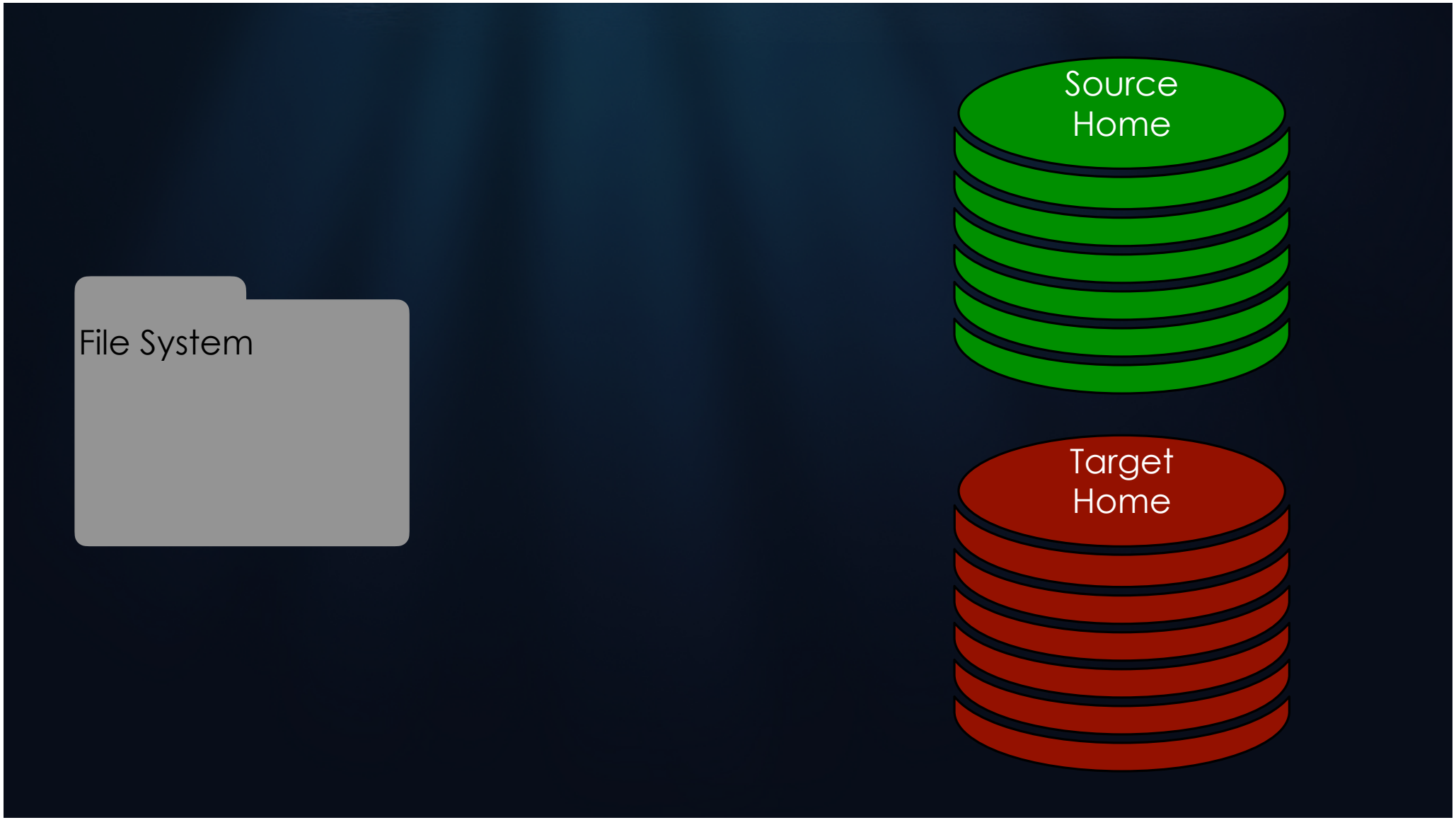
ORACLE

Building Upgrade Images



docker

ORACLE


File System
Source Home
Target Home


Create Target Filesystem
Add Source Home
Add Target Home
Attach Target Home

For 18c + 19c targets, APEX must be removed or upgraded in the source database:

```
Upgrade Oracle Application Express (APEX) manually before the database upgrade.

The database contains APEX version 5.0.4.00.12. Upgrade APEX to at least
version 18.2.0.00.12.

Starting with Oracle Database Release 18, APEX is not upgraded automatically as
part of the database upgrade. Refer to My Oracle Support Note 1088970.1 for
information about APEX installation and upgrades.
```

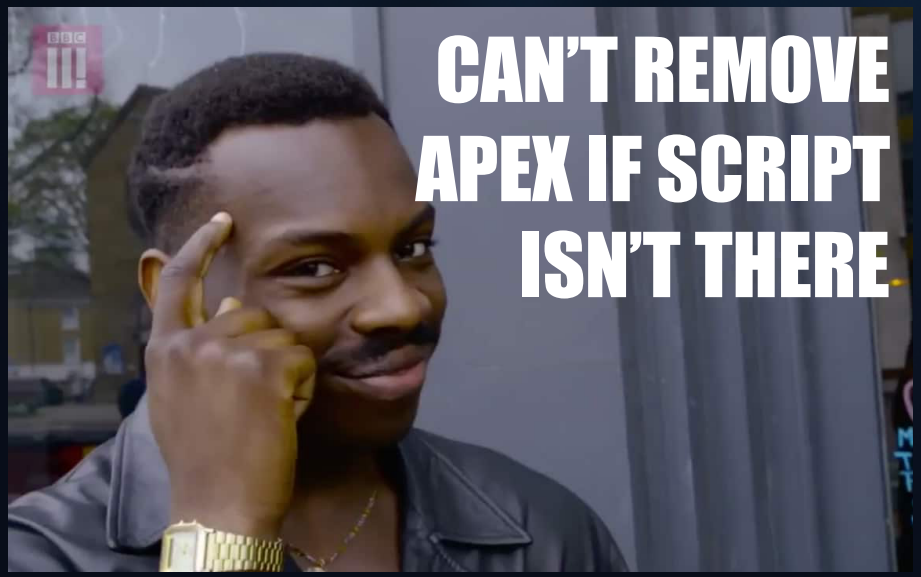
To remove APEX...

```
cd $ORACLE_HOME/apex
@apxremov_con.sql
```

Source and Target Images

`installDBBinaries.sh` *removes components* from ORACLE_HOME after DBCA runs.

Including APEX. :(



Remove this line

```
# Remove not needed components
# APEX
rm -rf $ORACLE_HOME/apex && \
# ORDS
rm -rf $ORACLE_HOME/ords && \
# SQL Developer
rm -rf $ORACLE_HOME/sqldeveloper && \
```

Source and Target Images

Do You PDB?

Official Oracle images on GitHub are PDB only!

Database creation performed by DBCA…

…using response files…

…created from templates in each version directory!

```
egrep -i "^createascontainer" */*.rsp.tmpl
12.1.0.2/dbca.rsp.tmpl:createAsContainerDatabase=true
12.2.0.1/dbca.rsp.tmpl:createAsContainerDatabase=true
18.3.0/dbca.rsp.tmpl:createAsContainerDatabase=true
19.3.0/dbca.rsp.tmpl:createAsContainerDatabase=true
```

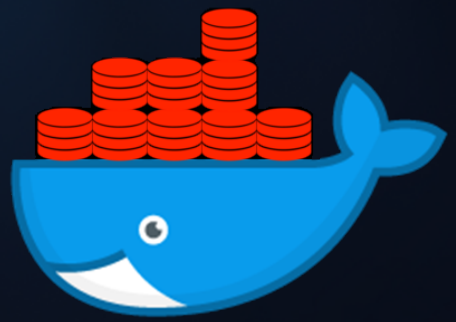
For non CDB, change to `false`

Source Image, Target Optional

Build the Source, Target Images

Run `.buildDockerImage.sh` as normal

Build the Working Image



docker

ORACLE

Completed: Source, Target Images

```
> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
oracle/database     12.1.0.2-ee         1f04faeab590        2 months ago        5.29GB
oracle/database     19.3.0-ee           df13428803ce        3 months ago        6.51GB
oraclelinux         7-slim              3f15c01b91bb        4 months ago        120MB
```

How do we upgrade this?

Create an Upgrade Image `Dockerfile`

Add a new **`ENV`** variable

```
FROM oraclelinux:7-slim as base

...

ENV TARGET_HOME=/opt/oracle/product/19c/dbhome_1
```

While we're here… add Helpful Things

`file-5.11`: Prerequisite for 19c preinstall

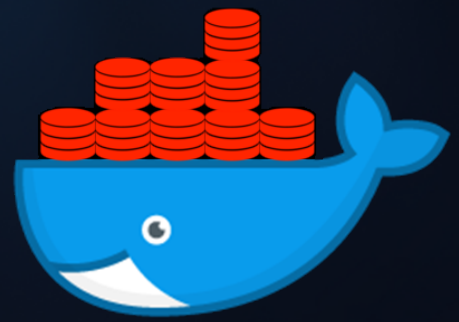
Amend this list to suit your needs and tastes

```
less                                tree
oracle-epel-release-el7             vi
strace                              which
```

`rlwrap`: Adds bash-like functionality to CLI

```
sync
yum -y install rlwrap
```

COPY Magic!



docker

ORACLE

Create an Upgrade Image `Dockerfile`

`COPY` the source `ORACLE_BASE` and target `ORACLE_HOME`

```
FROM base

# Source Base
COPY --chown=oracle:dba --from=oracle/database:12.1-ee \
                        $ORACLE_BASE $ORACLE_BASE
# Target Home
COPY --chown=oracle:dba --from=oracle/database:19.9-ee \
                        $TARGET_HOME $TARGET_HOME
```

Upgrade Image `Dockerfile`

Run the `root` scripts, including the target `root` script

Attach the target `ORACLE_HOME`

```
USER root
RUN $ORACLE_BASE/oraInventory/orainstRoot.sh && \
    $ORACLE_HOME/root.sh && \
    $TARGET_HOME/root.sh

USER oracle
RUN $TARGET_HOME/oui/bin/attachHome.sh
```

Upgrade Image `Dockerfile`

Optionally add 8080 to **EXPOSE**

The remainder is identical to the original `Dockerfile`(s)

```
WORKDIR /home/oracle

VOLUME ["$ORACLE_BASE/oradata"]
EXPOSE 1521 5500 8080
HEALTHCHECK --interval=1m --start-period=5m \
   CMD "$ORACLE_BASE/$CHECK_DB_FILE" >/dev/null || exit 1

# Define default command to start Oracle Database.
CMD exec $ORACLE_BASE/$RUN_FILE
```

Build the Upgrade Image

Run `docker build` from the `Dockerfile` directory

```
docker build -t oracle/database:12c-19c-ee .
```

Use `-f` to identify a non-standard `Dockerfile`:

```
docker build -t oracle/database:12c-19c-ee \
    -f Dockerfile.Name .
```

Tip: Create the `Dockerfile` in a new directory or add a `.dockerignore` file to prevent lengthy context builds

Completed: Source, Target and Upgrade Images!

```
> docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
oracle/database         12c-19c-ee          ea79f8b6081e        2 months ago        14.3GB
oracle/database         12.1.0.2-ee         1f04faeab590        2 months ago        5.29GB
oracle/database         19.3.0-ee           df13428803ce        3 months ago        6.51GB
oraclelinux             7-slim              3f15c01b91bb        4 months ago        120MB
```

Question: Why not install binaries “normally”?

```
docker cp /localdir/target_install.zip SOURCE:/opt/oracle
```

Then:

```
unzip /opt/oracle/target_install.zip etc.
```

Seems easier (and obvious)

Docker uses *union filesystems* (read only)

Modifying files & directories:

- Creates new layers
- Increases container size

Is there an easier way?

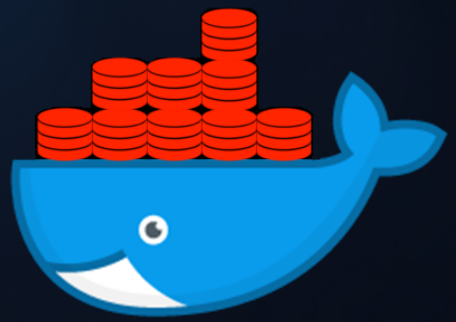
Yes:

```
https://github.com/oraclesean/docker-oracle
```

Oracle/Docker builds including:

- 11.2.0.4
- 19.6, 19.7, 19.8, 19.9
- CDB, non-CDB
- Upgrades
- Multiple/custom PDBs
- Dynamic builds
- Separate BASE, HOME, INV, ORADATA directories
- Data Guard
- Sharding

`docker run` an upgrade!



docker

ORACLE

Run the Container

```
CON_NAME=UPG

docker run -d \
       -v /data_dir/$CON_NAME:/opt/oracle/oradata \
       -e ORACLE_SID=UPG \
       -p 1921:1521 \
       -p 1980:8080 \
       --name $CON_NAME \
       oracle/database:12c-19c-ee

docker logs -f $CON_NAME
docker exec -it $CON_NAME bash
```

Use unique directories for oradata volumes

DON'T DO THIS!

```
docker run -d –name container1
       -v /data_dir:/opt/oracle/oradata imagename
docker run -d –name container2
       -v /data_dir:/opt/oracle/oradata imagename
```

The databases in these containers will share the same directory structure including control files and datafiles!

(That is a *Bad Thing*!)

Source Database Preparations

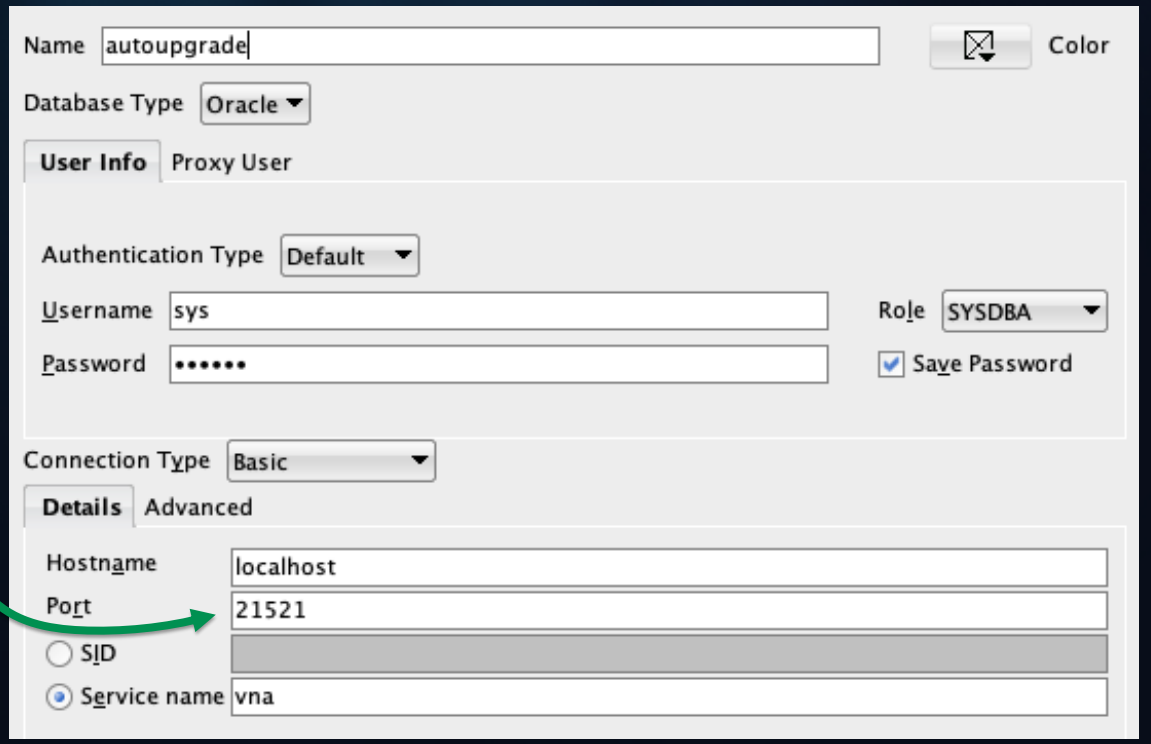
Set init parameters

Optionally start archivelog (required for AutoUpgrade):

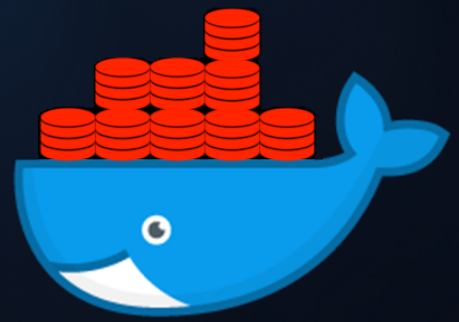
```
alter system set ... scope=spfile;
shutdown immediate
startup mount
alter database archivelog;
shutdown immediate
startup
```

Connect to SQL Developer

Local port mapped to container port 1521 by `docker run`



Run AutoUpgrade



docker

ORACLE

Run AutoUpgrade

Create a configuration file

```
cat /opt/oracle/autoupgrade/config.txt
global.autoupg_log_dir=/opt/oracle/autoupgrade
vna1.source_home=/opt/oracle/product/12.1.0.2/dbhome_1
vna1.target_home=/opt/oracle/product/19c/dbhome_1
vna1.sid=VNA
vna1.start_time=now
vna1.log_dir=/opt/oracle/autoupgrade/VNA
vna1.upgrade_node=855dea5a845e              # <- Container name
vna1.target_version=19.8
```

Run AutoUpgrade

```
$TARGET_HOME/jdk/bin/java -jar $TARGET_HOME/rdbms/admin/autoupgrade.jar \
                          -config /opt/oracle/autoupgrade/config.txt \
                          -mode analyze

$TARGET_HOME/jdk/bin/java -jar $TARGET_HOME/rdbms/admin/autoupgrade.jar \
                          -config /opt/oracle/autoupgrade/config.txt \
                          -mode deploy
```

Monitor AutoUpgrade

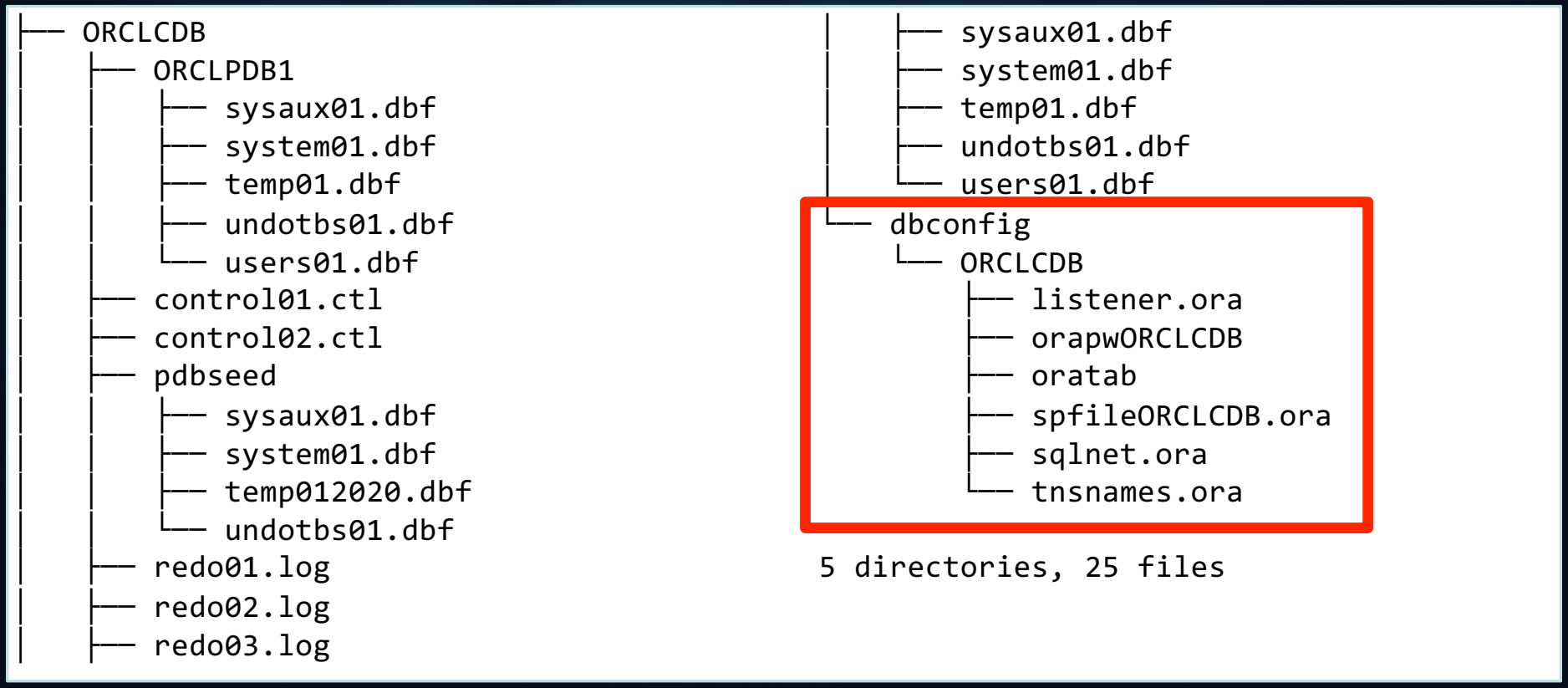
```
cd $ORACLE_BASE/autoupgrade/cfgtoollogs/upgrade/auto
nohup python -m SimpleHTTPServer 8080 &

http://localhost:1980/state.html
```

Local port mapped to container port 8080 by `docker run`

Tips and Tricks

Create Gold Images for Data



Create Gold Images for Data

```
docker run … -v /data_dir/$CON_NAME:/opt/oracle/oradata
```

`data_dir/$CON_NAME` contains the *entire database!*

- `data_dir/$CON_NAME/SID` = datafiles
- `data_dir/$CON_NAME/dbconfig/SID` = configurations

Create a gold image:

```
cp -r /data_dir/$CON_NAME /data_dir/gold
```

Recreate from the Gold Image

Remove the container and data:

```
docker rm $CON_NAME
rm -fr /data_dir/$CON_NAME
```

Restore the data and recreate the container:

```
cp -r /data_dir/gold /data_dir/$CON_NAME
docker run ...
     -v /data_dir/$CON_NAME:/opt/oracle/oradata ...
```

Clone from the Gold Image

Create a *new container* with existing Gold data:

```
cp -r /data_dir/gold /data_dir/CLONE

docker run -d
       -v /data_dir/CLONE:/opt/oracle/oradata \ # <- New dir
       -e ORACLE_SID=VNA \          # <- Same SID
       -p 2021:1521 \               # <- Change local port
       -p 2080:8080 \               # <- Change local port
       --name CLONE \               # <- New container name
       oracle/database:12c-19c-ee # <- Same image
```

Snapshots (PIT backups) from Containers

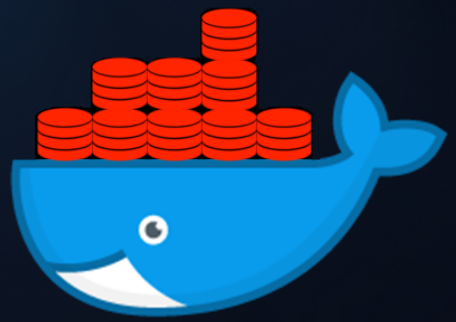
Create a snapshot as a new image:

```
docker commit $CON_NAME post_upgrade
cp -r /data_dir/$CON_NAME /data_dir/post_upgrade
```

Restore a snapshot:

```
docker rm -f $CON_NAME
cp -r /data_dir/post_upgrade /data_dir/$CON_NAME
docker run -d \
   -v /data_dir/$CON_NAME:/opt/oracle/oradata \
   -e ORACLE_SID=VNA ... # Same values used to create container
   --name $CON_NAME post_upgrade
```

Opportunities for Testing Upgrades on Docker



docker

ORACLE

Opportunities—Process

Compare different methods

Test backup and restore

Test GRP and Flashback

Update and correct documentation

Iterate and perfect process

Validate assumptions

Explore new features

Opportunities—Performance

Use Data Pump to export, import statistics

- Compare before/after plans
- Get comfortable with new optimizer options
- Determine changes for immediate implementation
- Develop action plans to:
  - Identify and isolate problems
  - React and resolve

Opportunities—Operations

Employ DevOps practices

- Establish metrics
- Build automation
- Create unit & functional tests

Opportunities—Operations

Develop and exercise tooling

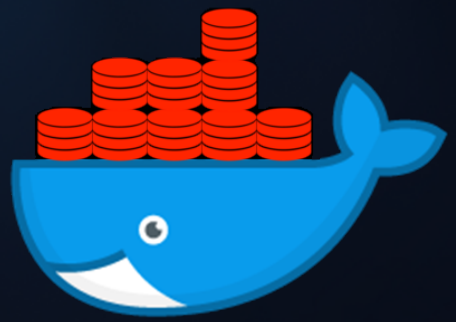
- Monitor
- Discover/observe patterns & anti-patterns
- Test observations
- Get smarter
- Add monitoring
- Repeat

Opportunities—Practice, Practice, Practice

Prepare like elite athletes and teams

- Build confidence in yourself, others
- Develop “muscle memory” for your upgrade
  - Respond instinctively
  - Improve reaction time
  - Know your tools and systems

Closing Thoughts



docker

ORACLE

MYTH: We have documentation for that

Is it current?
Is it accurate?

Documentation doesn't address:

Stress	Coordination	Research
Urgency	Phone calls	Management
Confusion	Messaging	Doubt, Panic
Multitasking	Alerts	Conflict

MYTH: Production is under change control

Older systems are more likely to contain undocumented or unexpected idiosyncrasies that introduce brittleness and fragility.

MYTH: Success in pre-prod assures success in prod

Production upgrades rarely go without incident.

The best way to prepare for the unexpected
is to make the fundamentals second-nature.

ORACLE

Questions

oraclesean.com

https://www.linkedin.com/in/soscott/

@oraclesean

https://github.com/oraclesean

sean.scott@viscosityna.com

Search "OracleSean" on YouTube