VISCOSITY
PROFESSIONAL SERVICES

ORAPUB
POWERED BY VISCOSITY



Oracle RAC – Does failover really work?

Gary Gordhamer

Oracle ACE Director
Managing Principal Consultant

@ggordham.bsky.social
linkedin.com/in/ggordhamer/
gary.gordhamer@viscosityna.com

NLOUG
Nov 13, 2026

NL.OUG
NL ORACLE USER GROUP



Gary Gordhamer

Oracle ACE Director
Managing Principal Consultant

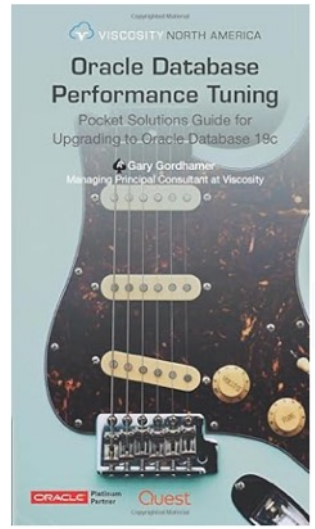
@ggordham.bsky.social
linkedin.com/in/ggordhamer/
gary.gordhamer@viscosityna.com

30+ years with Oracle technology

- Database 6.x – 23c
- Oracle Financials 7, E-Business Suite R11, R12.1, R12.2
- IDM integrations and Security hardening

Worked in many industries including

- manufacturing
- financial
- marketing
- healthcare
- utilities
- Government



Wrote the book:
“Oracle Database Performance Tuning: Pocket Solution Guide Series For Upgrading Oracle Databases”

Active member of the Oracle user community presenting frequently!

Oracle ACE Director

Viscosity's Oracle ACEs

The Oracle ACE Program

The Oracle ACE Program recognizes and rewards individuals for their contributions to the Oracle community.



Charles Kim
CEO | Co-Founder

@racdba
ACE Director



Rich Niemiec
Chief Innovation Officer

@richniemiec
ACE Director



Craig Shallahamer
Applied AI Scientist

@orapub
ACE Director



Sean Scott
Principal Consultant

oraclesean.com
ACE Director



Gary Gordhamer
Principal Consultant

ggordham.bsky.social
ACE Director



Marco Pereira
Software Architect

@markuspg
ACE Associate

viscosityna.com

@ViscosityNa

Want to try this out?

DEMO

You can download all the code from GitHub

https://github.com/ggordham/ora-presentations/tree/main/rac-failover-NLOUG

All code are ”as is” and meant for teaching purposes only.

DO NOT run these in any production system or system relied upon by your development teams.

Preventing Application Downtime

Infrastructure – High Availability

- Load balancer (user session)
- Redundant application servers
- Clustered database servers (Oracle RAC) ← This Discussion
- Redundant network + redundant storage fabric
- Highly redundant storage (SAN)
- Replicate data to second site (Oracle Dataguard)

Application changes – Edition Based Redefinition (different discussion)

- Application servers – require restarting app
- Database – changing objects causes waits / locks
- Need to synchronize DB objects and application code changes

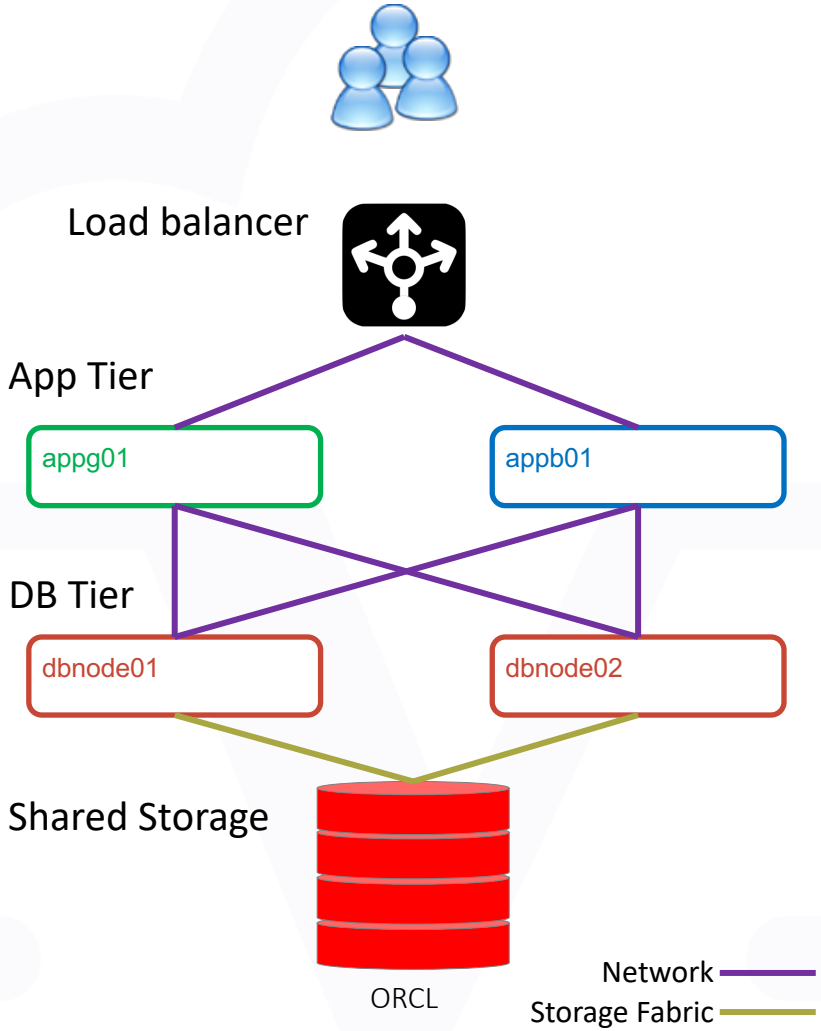
Simple HA application

- Fully redundant network
- Multiple physical servers for app and DB tiers
- Load balancer routes users to “living” app tiers
- Oracle database driver routes connections to living DB nodes
- Storage fabric is fully redundant

Each tier has multiple systems

- Change hardware, OS, software one at a time
- Load Balancer routes user sessions

What about the DB tier? RAC



Oracle Real Application Cluster (RAC)

Introduced in version 10g (10.1)

Based on Oracle Parallel Server (OPS)

Uses Oracle Grid Infrastructure (GI) or Oracle Cluster Ware (OCW)

Multiple physical or virtual servers to access the same database files

Benefits:

- Scale database to more CPU and Memory than a single server
- Fault tolerance for server or server component failures
- Allows for “on-line” maintenance of OS, DB software, and hardware

But does it really work?

What are your experiences?

Why are database changes hard?

ACID – keeping data consistent

- Atomic, Consistent, Isolated, Durable

App tier “waits” for database work

- DB connections are “sticky”
- Failures “break” the application
- May transfer large amounts of data between DB and app

Network - connects everything

- Fragile
- Tunned for “multipurpose”
- Small payload

What is the fix for the DB?

Oracle Transparent Application Continuity (TAC)

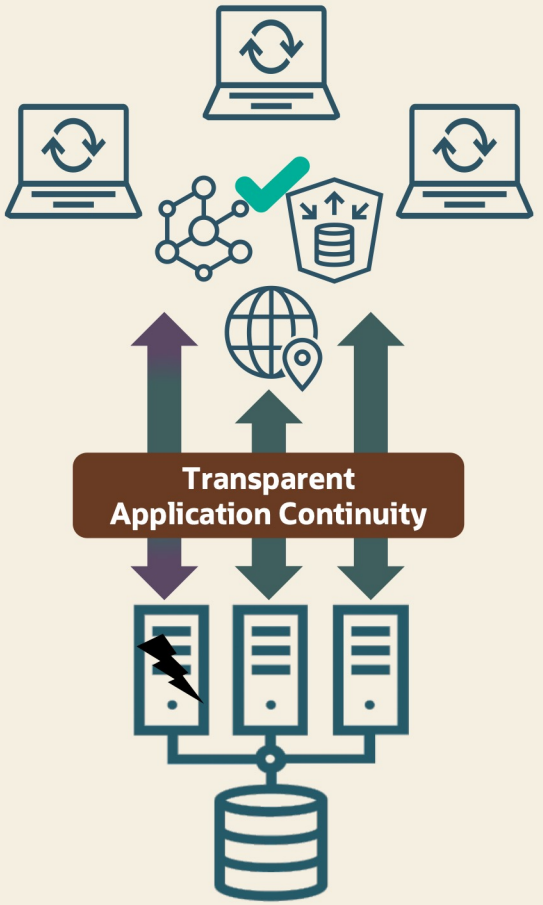
Why is it needed? Networking & TCP/IP

Transparent Application Continuity (TAC)

Keeps applications online during outages without requiring any custom error handling

- Hide downtime from end-users
  - TAC rebuilds the session state and replays in-flight transactions as part of automated session failover
- Eliminate errors unless unrecoverable
  - For both planned maintenance and unplanned outages
- Fast Application Notification (FAN) and session draining
  - FAN notifies clients of database status changes - helps break applications out of TCP timeouts
  - Draining causes sessions to complete their work on a given instance to prepare the node or the database for maintenance - no application changes required
- Best used with an Oracle (compatible) connection pool
- Oracle AI Database 26ai provides Application Continuity through DBMS_ROLLING operations.

New In 26ai



Copyright © 2025, Oracle and/or its affiliates | Public

TCP / IP (Internet Protocol Suite)

How data is addressed, packaged, transmitted, and routed

- Transmission Control Protocol (TCP) – Reliable, receive response (HTTP, SSH, SFTP, SQL NET)
- User Datagram Protocol (UDP) – High speed, unreliable (DNS, Ping, Cluster, etc.) *spit & forget*
- Internet Protocol (IP) – used by TCP / UDP



OSI model	DOD model	What we Use	Where its Configured	Layer
Application	Process / Application	HTTP(s), SSH, SFTP, DNS, SQL Net	Application	Packet Inspection
Presentation				
Session				
Transport	Host to Host	TCP / UDP	PORT	
Network	Internet	IP	IP Address	3
Data Link	Network Access	Ethernet / WiFi (ISDN / T1 / etc)	MAC Address	2
Physical				

Each layer is responsible for only their piece

IPv4 – the world we live in

IPv6 is readily available, but we pretty much just use IPv4

- Sets the maximum size of the data packet (64K, but limited by MTU EG. 1500)
- Sets maximum number of destination addresses 1.0.0.1 – 255.255.255.254
- Sets aside a number of ”special-use addresses”

*Note: data size is throttled by Maximum Transmission Unit (MTU) setting though **all devices** in the transmission must support set size, default is 1,500 bytes.*

`Addresses are four octets (4 bytes)`

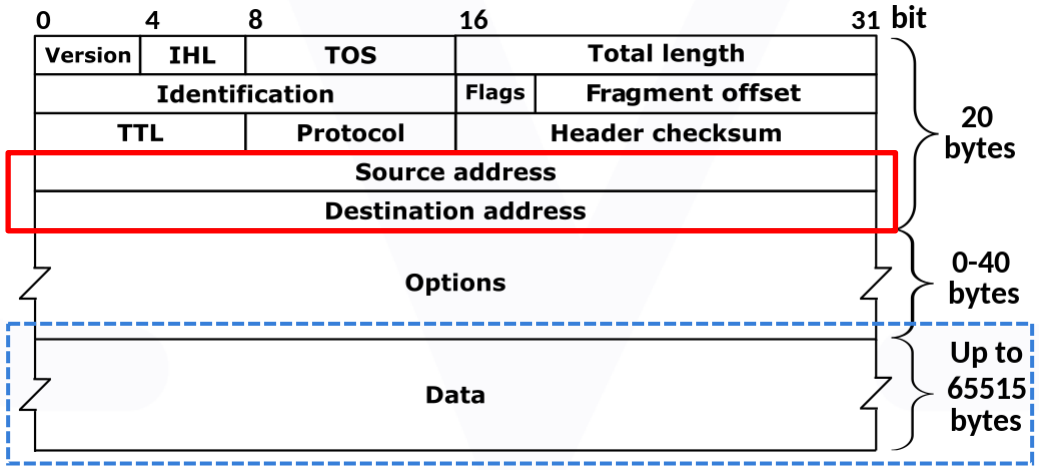
`11000000.10101000.000000.00001010`

`C0.A8.00.0A`

`192.168.0.10`

STATIC – IP does not change

DHCP – IP can change on each “renewal”



Subnets – how we route data

```
traceroute to www.google.com (172.217.4.68)
192.168.0.1       16.747 ms
65.31.113.122     34.817 ms
66.109.6.204      43.910 ms
72.14.222.220     34.542 ms
142.251.231.244   39.896 ms
108.170.243.174   29.339 ms
142.251.60.10     28.849 ms
216.239.51.117    46.739 ms
108.170.243.165   24.019 ms
216.239.51.117    48.676 ms
172.217.4.68      42.191 ms
```

SUBNET Mask defines the size of the subnet

(**CIDR** or Classless Inter-Domain Routing)

255.255.255.0 – Netmask – first three octets match

AKA a /24 subnet or CLASS C subnet (*IP Calculator*)

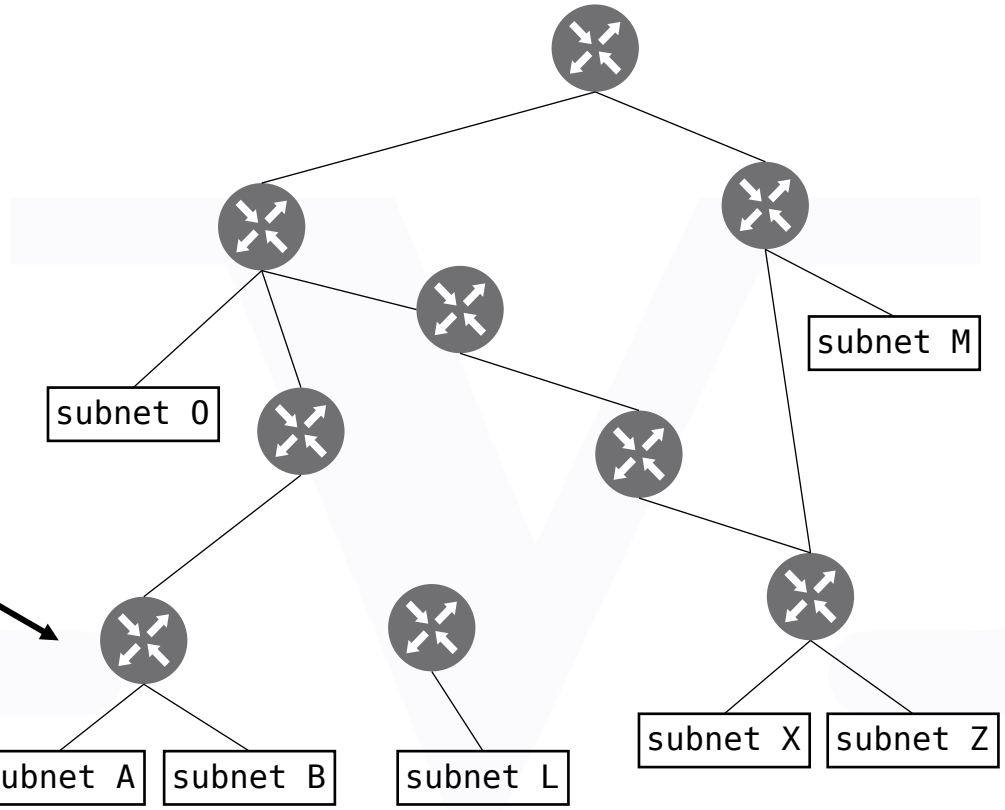
- Three octets (3x8 = 24)
- 192.168.1.1 – 192.168.1.254 (254 addresses)
- E.G. /28 = 192.168.1.129 – 142 (16 addresses)
  Netmask 255.255.255.240

You can have many sizes of networks

- CIDR size is critical when setting up cloud tenancy
- Subnets must be unique in a route!
- Multiple paths are allowed

Routers move data between networks

Note: TCP/IP timeouts are based on concept of traversing multiple subnets (hence 30sec or 60sec)

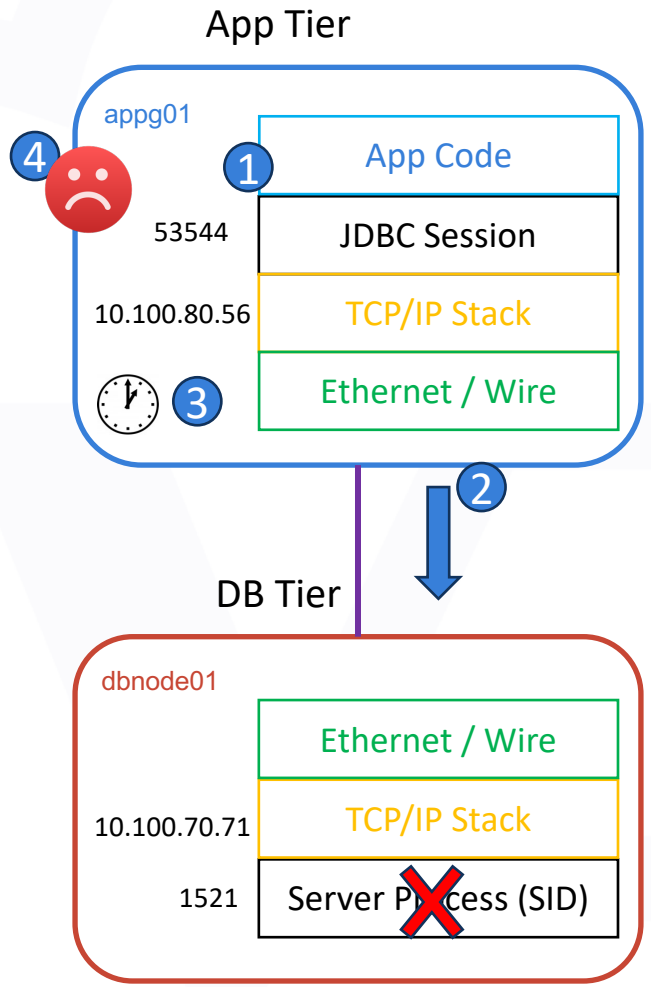


Failure - without TAC

1. App tier – JDBC sends packet to DB
   could be start of a statement
   could be request data during a statement
   could be finish or commit
2. Packet handed over TCP/IP to Ethernet
   App tier ”waits” for response
3. Wait time is set at TCP/IP level, default 60 seconds
   JDBC session waits for response (long time….)
4. Eventually JDBC returns an error
   application gets error
   code in app would have to deal with error

Outcome: Application hangs and then fails
Why not decrease timeout?
Affects communication with everything
Each layer (see slide 11) is responsible for itself

APP code would have to catch and deal with any connection errors!



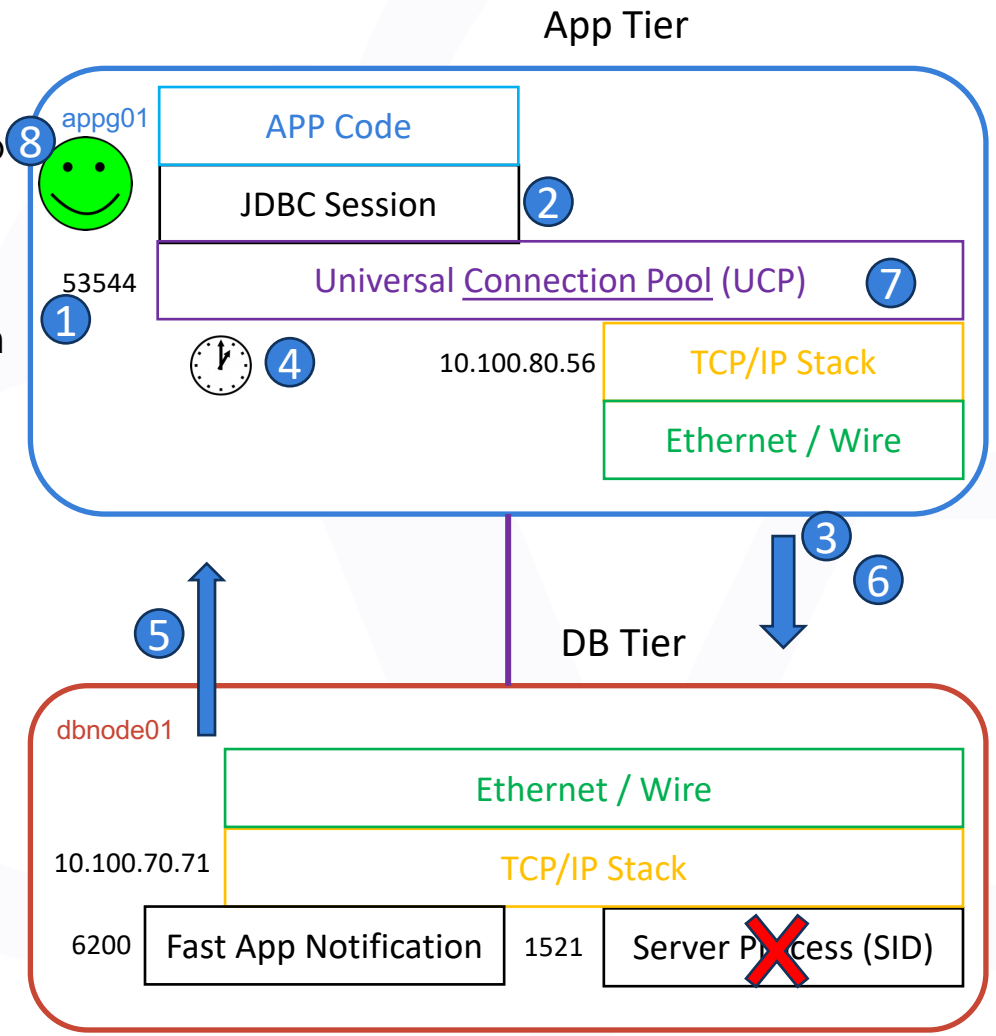
Failure - with TAC

1. APP tier – Connection Pool opens a set of sessions to the database.
2. APP code requests a connection from pool
3. Pool sends request to DB using available connection
4. Pool manages timeout waiting for response
5. Out of band, Fast Application Notifications (FAN) sends updates if a service or node fails
6. Out of band, UCP re-establishes connections to DB
7. If time out expires or FAN notification is received due to error, UCP replays the transaction on a different connection

Outcome: Application may hang, but eventually finished

APP code does not have to deal with connection error!

You can control timeout settings in the connection pool!



Demonstrations

Very simple code, from Oracle examples and heavily modified

- Creates connection pool
- Starts multiple application threads
- Each thread does simple inserts and selects
- One thread can do long query every few minutes (parameter)

Run demo through a few situations

Situation	No TAC	TAC
Kill single DB connection	Errors received	No errors
DB server killed	All sessions fail (crash)	No errors
Move service (planned maintenance)	No errors * (small delay in ms)	No errors (small delay in ms)

Settings used

JDBC Setting	No TAC	TAC
setConnectionFactoryClassName (JDBC driver)	oracle.jdbc.pool.OracleDataSource	oracle.jdbc.replay.OracleDataSourceImpl
setFastConnectionFailoverEnabled	FALSE	TRUE
setValidateConnectionOnBorrow	FALSE	TRUE
setConnectionWaitDuration	5	5

DB Service Setting	No TAC	TAC	Notes (setting set with srvctl)
failovertype	NONE	AUTO	[NONE | SESSION | SELECT | TRANSACTION | AUTO] to enable AC (TRANSACTION) or TAF (SELECT | SESSION) or TAC (AUTO)
failover_restore	NONE	AUTO	[NONE | LEVEL1 | AUTO] session state restore before replay TAF (LEVEL1) or TAC (AUTO)
commit_outcome	FALSE	TRUE	[TRUE | FALSE] Enables commit durability / Transaction Guard

Note: any other service setting for TAC is deprecated

TNS connection -

TNS Connection Setting	No TAC	TAC	
CONNECT_TIMEOUT	90	90	Time to wait for database connection to establish defaults to seconds and forever (ms, sec, min)
RETRY_COUNT	50	50	Number of times address list is run through before abandoning connection attempt
RETRY_DELAY	?	3	Time between new connection retry attempts, defaults to seconds (ms, sec, min)
TRANSPORT_CONNECT_TIMEOUT	3	3	How long to wait for the transport to establish connection (TCP / IP) defaults to seconds (ms, sec, min)
RECV_TIMEOUT	N/A	20	How long to wait for data from peer after connection established, defaults to forever (ms, sec, min)

Note: settings can be put in TNS entry instead of setting at server / client level

```
jdbc:oracle:thin:@(DESCRIPTION=(RECV_TIMEOUT=20)(CONNECT_TIMEOUT=90)(RETRY_COUNT=50)
(RETRY_DELAY=3)(TRANSPORT_CONNECT_TIMEOUT=3)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)
(HOST=crs01-scan.gg.wi.rr.com)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=hr_tac)))
```

Note: JDBC URL for TAC enabled wrapped for display

Videos

Each demo is run once with NO TAC features, second with TAC enabled

- Demo 1 – Failover when session is killed
  like network failure / network interface flapping
- Demo 2 – Move service manually
  Process used for planned maintenance
- Demo 3 – Node down
  Hardware failure, power outage, etc.

For each we try to show

- Settings used
- SQL errors
- FAN notifications
- Replayed sessions

What code changes are needed?

Java Example, but similar change for any language.

```
// Create pool manager and data source
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
UniversalConnectionPoolManager poolManager =
        UniversalConnectionPoolManagerImpl.getUniversalConnectionPoolManager();

// Create and start the pool
poolManager.createConnectionPool((UniversalConnectionPoolAdapter)pds);
poolManager.startConnectionPool('my_pool_nanme');
```

Using the pool should look just like current code without a pool

```
// Request a connection
Connection c = pds.getConnection();

// Create statement and run it
PreparedStatement pstmt = c.prepareStatement("SELECT ENAME from EMP4AC WHERE ROWNUM < 10 and EMPNO = 1282");
ResultSet rs = pstmt.executeQuery();

// close statement and connection
pstmt.close()
c.close()
```

Notes on DataSource

When you create you connection pool you specify your JDBC driver.

- UCP supports any driver (MySQL, MS-SQL, DB2, etc.)
- We will focus on Oracle here

Oracle JDBC drivers

- `oracle.jdbc.pool.OracleDataSource` – Recommended minimum usage today, provided getPoolDataSource for UCP.
- `oracle.jdbc.replay.OracleDataSourceImpl` – Replay data source, recommended for all RAC recommendations.
- `oracle.jdbc.xa.client.OracleXADataSource` – Specific data source for XA transaction type usage (if you know, you know)
- `oracle.jdbc.driver.OracleDriver` – Legacy driver, not recommended for any current usage.

```
pds.setConnectionFactoryClassName("oracle.jdbc.replay.OracleDataSourceImpl");
```

Pool Settings – Critical!

Pool size

- `pds.setInitialPoolSize(ucpPoolSize);`
  // default 0 - connections created when the pool is created
- `pds.setMinPoolSize(ucpPoolSize);`
  // default 0 - the number of available + borrowed connections the pool tries to return to
- `pds.setMaxPoolSize(ucpPoolSize);`
  // default Integer.MAX_VALUE - the total number of available + borrowed connections

Pool Connection settings

- `pd.setConnectionValidationTimeout(55);`
  // default 15 - how many seconds to wait for connection to validate
- `pds.setConnectionWaitTimeout(10);`
  // default 3 - wait time in seconds for connection if all are busy
- `pds.setInactiveConnectionTimeout(60);`
  // default 0 - time in seconds before idle connection is closed (not borrowed)

Failover settings

- `pds.setFastConnectionFailoverEnabled(true);`
  // default disabled – enabled Fast Connection Failover (FCF / FAN) for the pool
- `pds.setValidateConnectionOnBorrow(true);`
  // default disabled – checks JDBC API isValid before trying to use a connection
- `pds.setSecondsToTrustIdleConnection(5);`
  //default 0 – skip connection validation if used within this number of seconds

Statement caching

- `pds.setMaxStatements(10);`
  // default 0 - number of statements to cache for reuse in each connection

Connection Pool – Do's & Don'ts

Don't oversize your connection pool

- Too many connections that are not used – wastes resources on app tier and DB tier

Don't keep connections open forever

- Allowing connections to close frees up resources on both app and DB tier
- Less things to "manage" during planned maintenance activity
- Cleans up – clears out open cursors, memory cache, etc.

Don't just accept the defaults

- Look at the parameters and adjust to match your needs / environment

Do make these parameters variables (parameter file)

- Using servlet container – adjustable normally in the container (Tomcat, Weblogic, etc.)
- Variables allow for easy modification when scaling out (adding tiers)

Do adjust over time

- Monitor usage / performance
- Adjust (down and up) as you need
- As you add tiers, change settings. More tiers, each tier may need less sessions

Pool Settings – Advanced

Clean connections after amount of use

- `pds.setMaxConnectionReuseTime(300);`
  // default 0 - close connection after this many seconds
- `pds.setMaxConnectionReuseCount(100);`
  // default 0 - close connection after this many borrows

Harvesting - not timeout

- `pds.setConnectionHarvestTriggerCount(2);`
  // default Integer.MAX_VALUE - when number of avaialble connections hits this number, harvesting will begin
- `pds.setConnectionHarvestMaxCount(5);`
  // default 1 - number of borrowed sessions to harvest when harvesting

Connection timeout reclaim

- `pds.setAbandonedConnectionTimeout(10);`
  // default 0 - reclaim borrowd connections if not used in this many seconds (rolls back)
- `pds.setTimeToLiveConnectionTimeout(18000)`
  // default 0 - seconds a connection can be borrowed before being reclaimed (rolls back)
- `pds.setQueryTimeout(60);`
  // time in seconds the pool waits for a query to finish executing, throws exception
- `pds.setTimeoutCheckInterval(60);`
  // default 30 - how frequent in seconds timeout properties are enforced (can delay timeouts)

Note on Oracle JDBC jar files

Multiple jar files in ORACLE_HOME/jdbc/lib and download page.

How to choose?

- Related to Java version and JDBC features
- ojdbc8.jar – JDBC 4.2 features, for JDK 8 and 11
- ojdbc10.jar – JDBC 4.2 features, for JDK 8 and 11
- ojdbc11.jar – JDBC 4.3 features, for JDK 11
- ojdbc17.jar – JDBC 4.3 features, for JDK 17 and above

There are additional JAR files required for additional SQL*Net features like SSL / TLS etc. See the manual for more information:

https://docs.oracle.com/en/database/oracle/oracle-database/26/jjdbc/introducing-JDBC.html

Generally – always use the newest version of drivers that you can!

TAC support

- Oracle JDBC Replay Driver 18c or later. This is a JDBC driver feature provided with Oracle Database 18c for Application Continuity.
- Oracle Universal Connection Pool (UCP) 18c or later.
- Oracle WebLogic Server 18c, and third-party JDBC application servers using UCP
- Java connection pools or standalone Java applications using Oracle JDBC - Replay Driver 12c or later with Request Boundaries
- OCI Session Pool 19c or later
- SQL*Plus 19c or later
- ODP.NET Unmanaged Provider 18c or later
- Oracle Call Interface based applications using the 19c OCI driver or later

Questions?

References

- JDBC / UCP downloads:
  https://www.oracle.com/database/technologies/appdev/jdbc-downloads.html
- ODP.NET / OCMT downloads:
  https://www.oracle.com/database/technologies/net-downloads.html
- ODBC / Instant client downloads Windows:
  https://www.oracle.com/database/technologies/instant-client/winx64-64-downloads.html
- Universal Connection Pool Developer's Guide
  https://docs.oracle.com/en/database/oracle/oracle-database/26/jjucp/intro.html
- Transaction Guard (Database Development Guide)
  https://docs.oracle.com/en/database/oracle/oracle-database/26/adfns/transaction-guard.html
- UCP / Replay statistics Java guides
  https://docs.oracle.com/en/database/oracle/oracle-database/26/jjuar/oracle/ucp/UniversalConnectionPoolStatistics.html
  https://docs.oracle.com/en/database/oracle/oracle-database/26/jajdb/oracle/jdbc/replay/ReplayStatistics.html
- Pool Data Source Java guide
  https://docs.oracle.com/en/database/oracle/oracle-database/26/jjuar/oracle/ucp/jdbc/PoolDataSource.html
- Application Continuity for JDBC Applications
  https://docs.oracle.com/en/database/oracle/oracle-database/26/jjdbc/application-continuity.html

References

- Fast Application Notification (FAN) – Whitepaper
  https://www.oracle.com/technetwork/database/options/clustering/overview/fastapplicationnotification12c-2980342.pdf
- SimpleFAN: Extending Oracle RAC High Availability to Custom Connection Management (blog)
  https://blogs.oracle.com/developers/simplefan-extending-oracle-rac-high-availability-to-custom-connection-management
- Real Application Cluster Administration and Deployment Guide
  https://docs.oracle.com/en/database/oracle/oracle-database/26/racad/server-control-utility-reference.html
- High Availability Overview and Best Practices
  https://docs.oracle.com/en/database/oracle/oracle-database/26/haovw/configuring-level-3-mask-unplanned-and-planned-failovers-applications.html
- Using transparent application continuity with current Java frameworks – this is how it works (blog)
  https://blogs.oracle.com/coretec/using-transparent-application-continuity-with-current-java-frameworks-this-is-how-it-works



VISCOSITY

PROFESSIONAL SERVICES