DOAG

Automate Database Deployments with SQLcl

Robert Marz
Sabine Heimsath

Databees.

Robert Marz – Independent Consultant

Primary Role

- Senior Technical Architect
- with database centric view of the world

DOAG (German Oracle User Group)

- Active Member of Database Community
- Responsible for Cloud Topics

@RobbieDatabee

robbie.databee.org

robert.marz@databee.de

Databees.

ORACLE ACE

Sabine Heimsath – Independent Consultant

Self-employed

PL/SQL and APEX Developer

Passionate Data Modeler

Active Member of
DOAG Development Community
(Responsible for PL/SQL Topics)

ORACLE ACE

Databees.

500+ Technical Experts Helping Peers Globally

ORACLE ACE PROGRAM

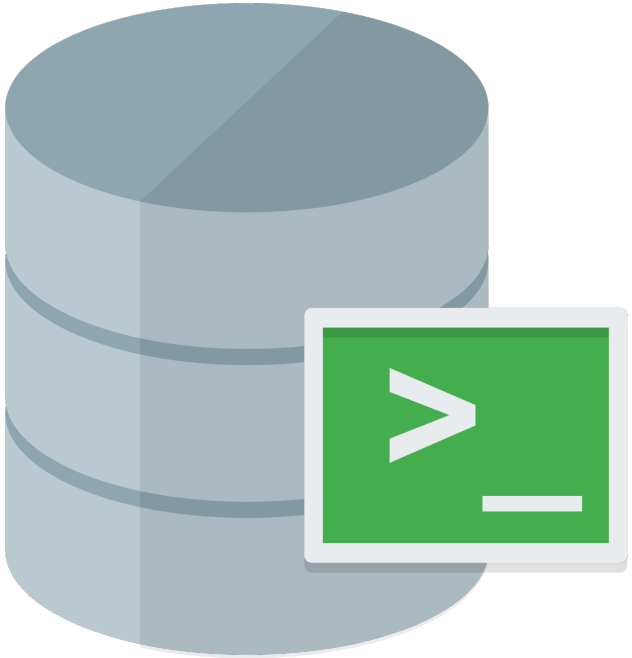
---

bit.ly/OracleACEProgram

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

SQLcl – An Overview

What is SQLcl?



Oracle SQL Developer Command Line

New	SQL*Plus
	Java Based
Modern	Command Line
	No Oracle Client needed
Included	$ORACLE_HOME/bin (12cR2)
	sqldeveloper/bin

SQLcl Usability Enhancements

Usability:

- Tab-Completions
- Change Directory
- History
- Command Editor

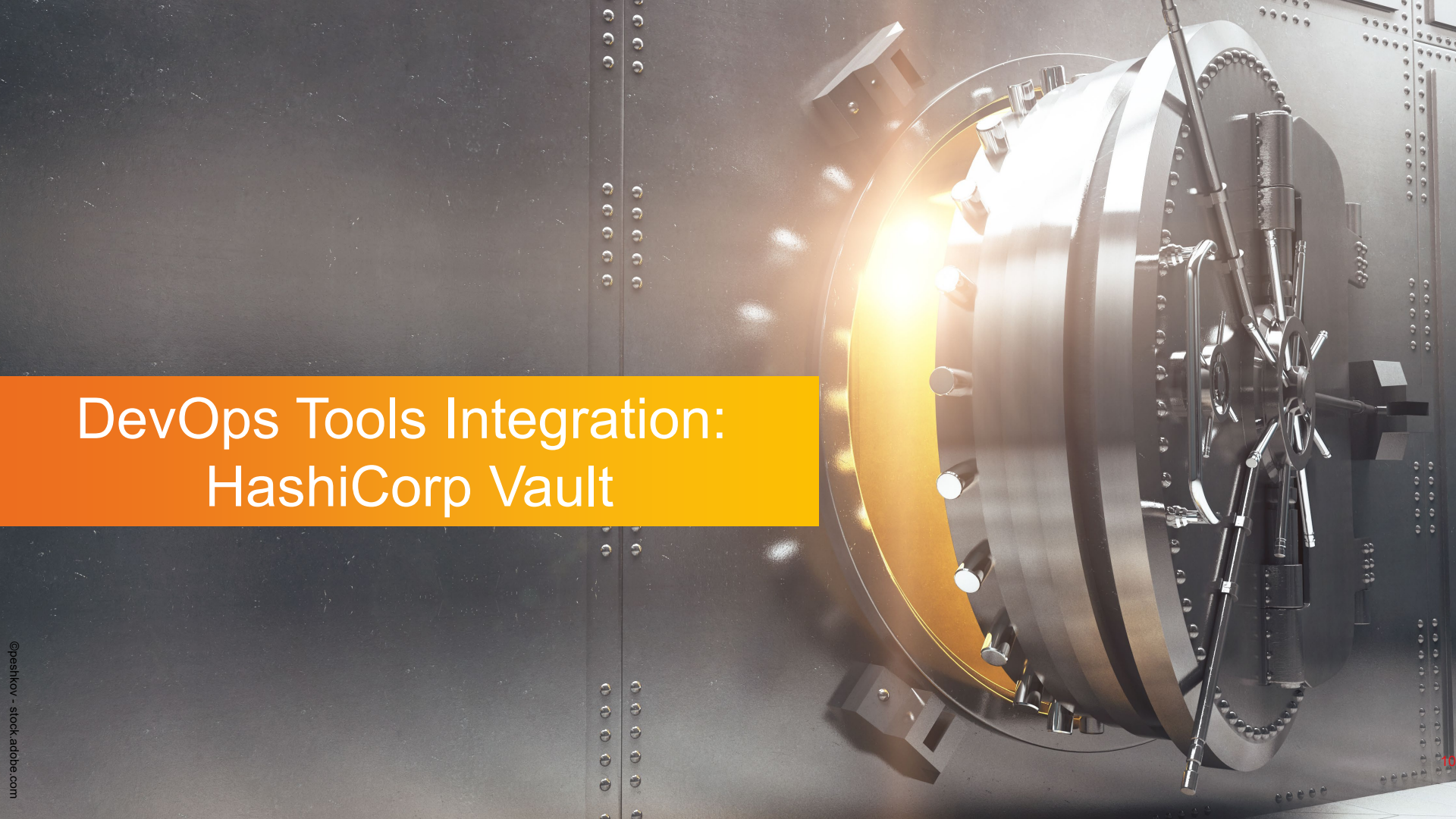
SQLcl Features

©BillionPhotos.com - stock.adobe.com

New Commands

cd
script
alias
apex
ddl
history
lb (liquibase)
rest
soda
vault
which
[…]


DEMO
delete
©niroworld - stock.adobe.com



DevOps Tools Integration: HashiCorp Vault

Keeping Secrets in Applications

Secrets

- Usernames
- DB Credentials
- API Tokens
- TLS Certificates

Secret Sprawl

- Source code
- Configuration Files
- Version Control System

Applications leak Secrets

- Logs
- Diagnostics
- Monitor

HashiCorp Vault: Concepts & Features

Vault Centralizes Secrets

Encrypts	at rest
	in transit
Access Control Lists	
Audit Usage	
Dynamic Secrets	ephemal (eg. 5 days)
	Unique, Revokable
Encrypt as a Service	Named Keys; Key Lifecycle
	High Level API (Enrypt, Decrypt - Sign, Verify)

HashiCorp Vault: Architecture

Client / Server

- Cluster / HA capable by default
- REST API (JSON Documents) / CLI

Open Source hosted on GitHub

- Released as single binary
- **`vault / vault.exe`**

Highly Pluggable / Extendable

- Authenticate
- Audit
- Store
- Secret Backends

HashiCorp Vault - SQLcl Integration

```
$ sql orcl/robbie

SQLcl: Release 19.2.1 Production on Sun Nov 03 20:37:10 2019

Copyright (c) 1982, 2019, Oracle.  All rights reserved.

Attempting vault based connection.
        Connecting to : robbie/xxxx@jdbc:oracle:thin:@127.0.0.1:1521/orcl
Connected using : http://127.0.0.1:8200/v1/cubbyhole/orcl/robbie
Connected.
```

```
SQL> help vault

 The vault command provides integration with HashiCorp Vault for saving and sourcing connection information.

 Vault configuration settings are read from the environment:
 VAULT_TOKEN, VAULT_ADDR

 SSL Settings are used from Java configuration.

vault config                 : Print current configuration
vault base                   : Sets base path                          Default : /cubbyhole/
vault list   [filter]        : Lists stored connections
vault store  [name]          : Store current connection to vault.      Default : 'global_name/username'
vault delete [name]          : Delete store name configuration.
vault autostore [true/false] : Sets auto storing of connections.       Default : false
SQL>
```

Enable: Environment Variables

- VAULT_ADDR
- VAULT_TOKEN

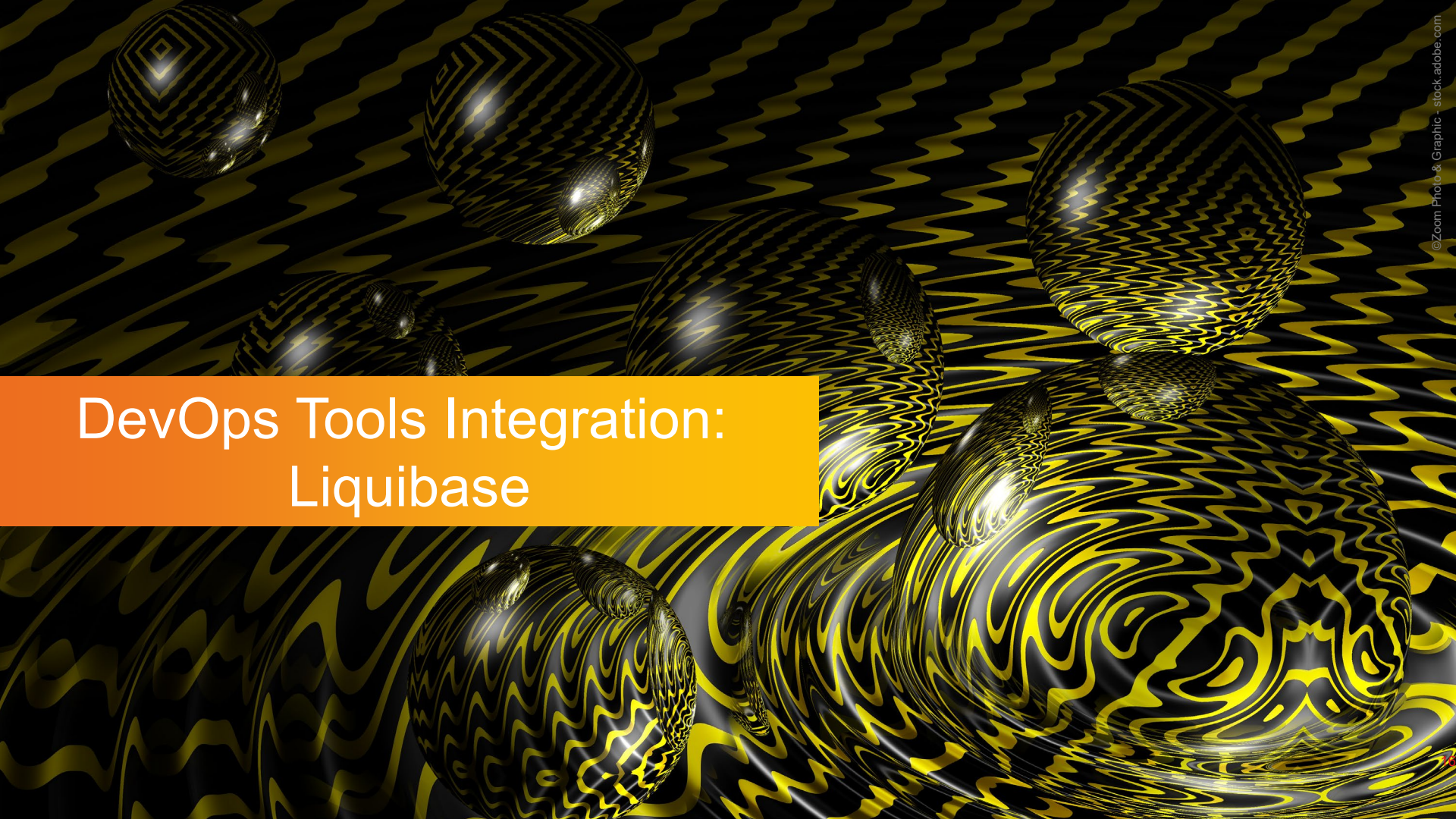
Connect Strings

- interpreted as vault KV Paths
- SQLcl attempts to read credentials
- if unsuccessful, use as normal connect string

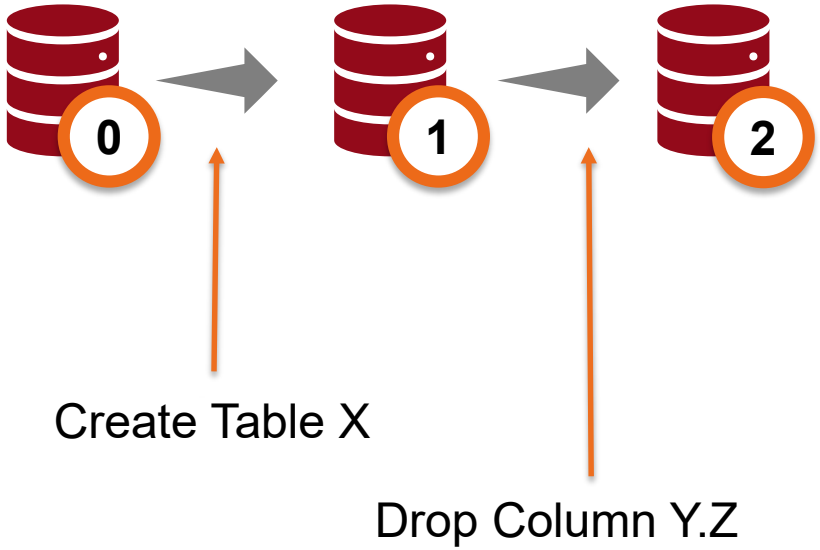
Established Connections

- Can be stored in vault
- Autostore available


DEMO
delete
©niroworld - stock.adobe.com


DevOps Tools Integration:
Liquibase
©Zoom Photo & Graphic - stock.adobe.com

Liquibase – Source Control for your Database



DB Changes

- Deploy, Rollback
- Track, Manage, Document
- DDL, Sources, Data

Java-based Open Source

- Community: liquibase.org
- Commercial: liquibase.com

No extra installation needed

- Liquibase libraries come with SQLcl

Liquibase

Supported Databases

- Oracle
- MySQL
- PostgreSQL
- SQL Server
- DB2
- more

Oracle features in SQLcl Liquibase

- Virtual Columns – ok
- Materialized Views – ok
- Hidden Columns – ?
- Partitions – ?
- Global Temporary Tables – ?

Liquibase – File Formats

```
<databaseChangeLog>
  <preConditions>
            ...
  </preConditions>

  <changeSet id="A" author="Shaun">
    <createTable tableName="persons">
            ...
    </createTable>
  </changeSet>

  <changeSet id="B" author="Shirley">
    <insert tableName="persons">
        ...
    </insert>
  </changeSet>
</databaseChangeLog>
```

Incremental Changes

- Change sets/Change logs
- File formats:
  - YAML
  - JSON
  - XML ← SQLcl
  - SQL
- Changelog Tables inside DB


DEMO
delete
©niroworld - stock.adobe.com

Scripting in SQLcl

Keyword script

executes

File
```
script scriptname.js
  -- (single line)
```

Inline Script
```
script            -- (linebreak)
  # type your script on the fly
```

Supported Languages

All JSR-223 Implementations

JavaScript (nashorn)

Python (Jython)

… lots more …

Endless Possibilities

controlling SQLcl Buffer

background DB Connections

… lots more …

Oracle DB Tools GitHub Examples

Globals

There are a few globals pushed into the scripting engine for use.

args -This is a simple array of the arguments passed along

Example:

```
for(var arg in args) {
   ctx.write(arg + ":" + args[arg]);
   ctx.write("\n");
}
```

sqlcl - This is SQLCL itself

```
setStmt(<String of stuff to run>)
          This can be a single statement, an entire script of stuff, or any sqlcl command such as "@numbers.sql"
```

```
run()
       Runs whatever is set via the setStmt function
```

Example:

```
/* Run any amount of command in the sqlcl prompt */
sqlcl.setStmt("select something from somewhere; @myscript \n begin null;end;");

sqlcl.run();
```

ctx (this has tons of methods but this is the single most important)

```
write(<String>)
```

Example:

```
ctx.write('Hello World');
```

util (again tons of methods)

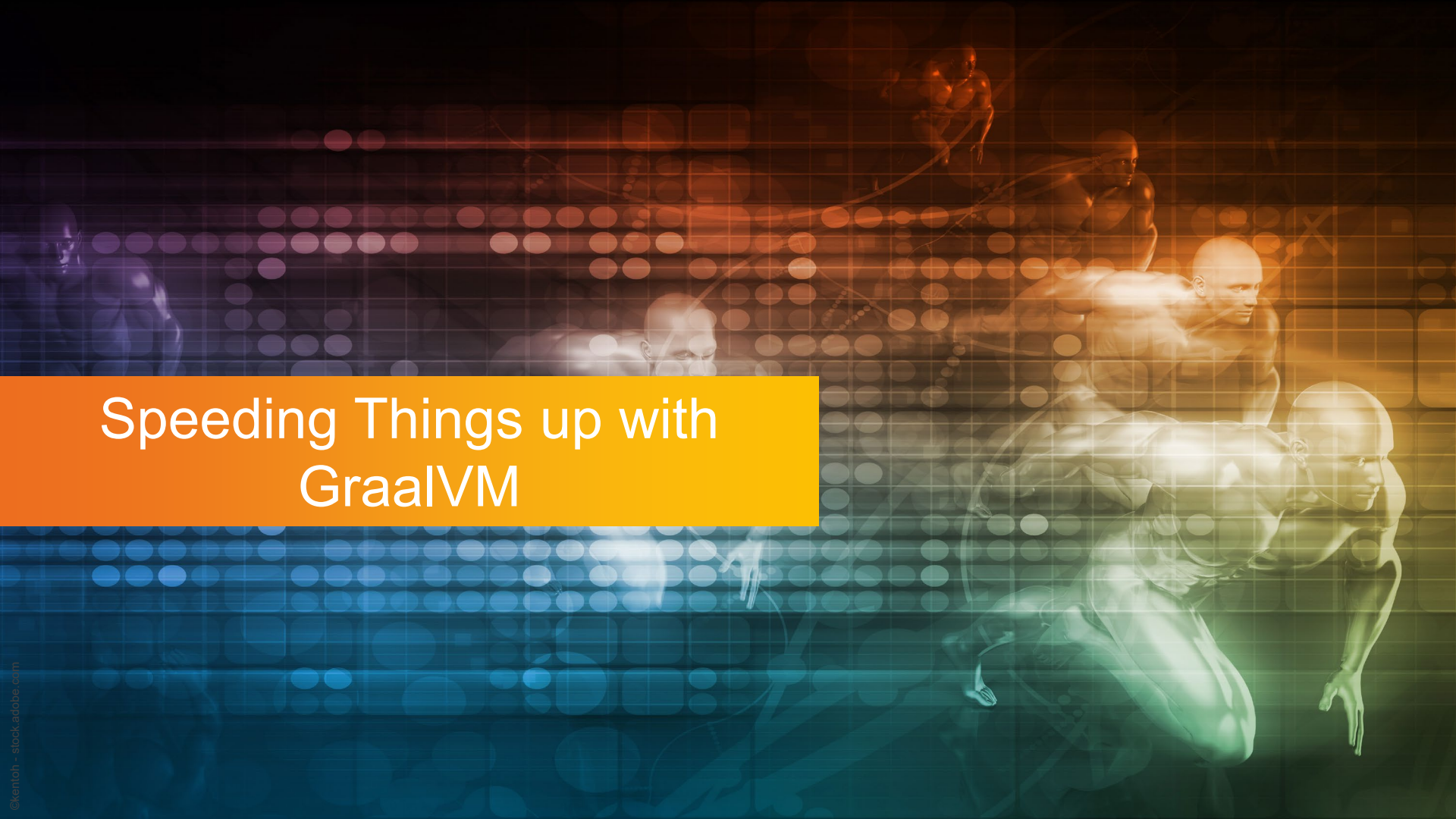
```
execute(<string>,binds)
      executes whatever is passed in with a boolean return for success/failure
```

```
executeReturnOneCol(<string>,binds)
      executes and returns the first row , first column
```

```
executeReturnListofList(<string>,binds)
      executes and returns an array(rows) of arrays(row).
```

```
executeReturnList(<string>,binds)
      execute and returns and array ( rows ) of objects ( row )
```


DEMO
delete
©niroworld - stock.adobe.com



SQLcl & GraalVM

Universal VM

JS, Python, Ruby, R
Java, Scala, Groovy, Kotlin, Clojure

Polyglot

Embed different Language

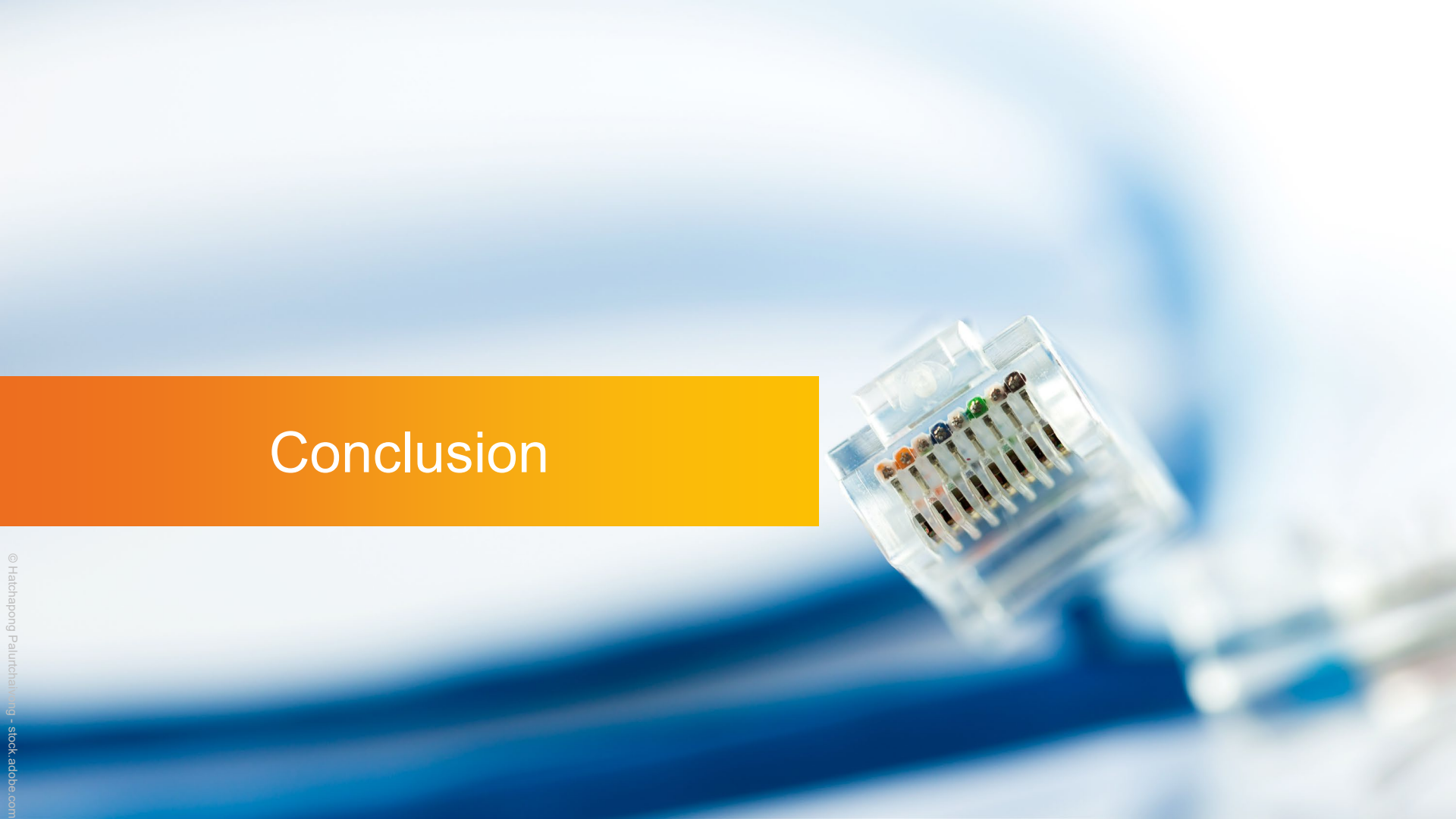
Execute SQLcl in GraalVM

Unpack Graal
set PATH & JAVA_HOME Variables
Lightning fast

Scripting

uncharted territory
very promising

Conclusion



Automate Database Deployments with SQLcl

- SQLcl – a mighty Tool
- Endless Possibilties
- Uncharted Territory