Hy APEX World_

DevOps Made Easy!

with Oracle Autonomous Database using PL/SQL and Git

Timo Herwix, Senior Consultant

code of change

Hyand by GOD | MT

Who am I?

Timo Herwix

Senior Consultant at MT | Part of Hyand since 2019

Previously worked as a Data Warehouse Developer

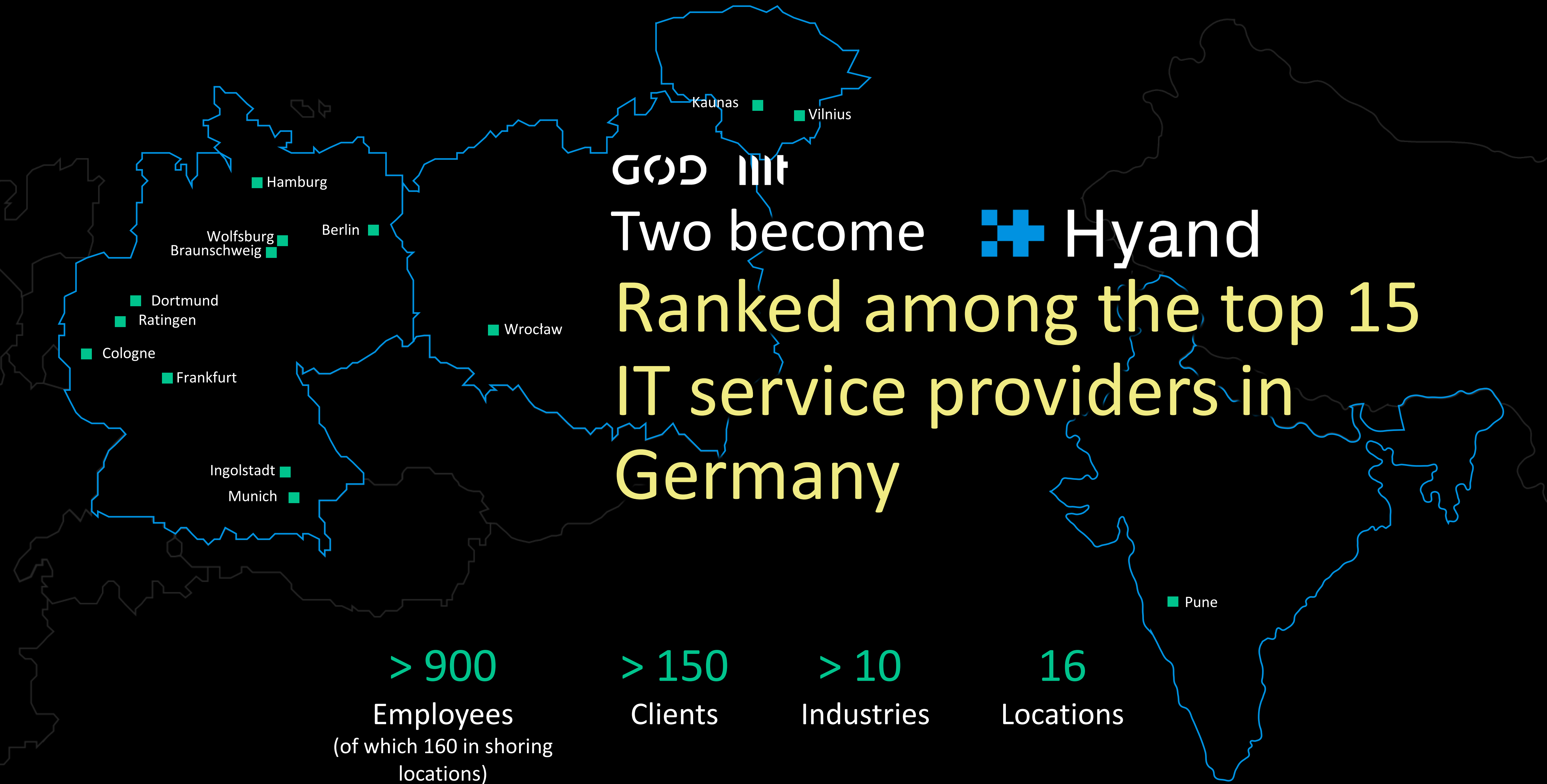
Oracle APEX since 2016

Oracle Databases since 2008

Blog author, conference speaker

Born in 1983, two children and living in Germany

Oracle ACE Associate



Two become Hyand

Ranked among the top 15 IT service providers in Germany

> 900	> 150	> 10	16
Employees (of which 160 in shoring locations)	Clients	Industries	Locations

Hyand

Agenda.

1

Introduction

2

Let´s dive deeper

3

Wrap-up

Hyand

DevOps Made Easy!

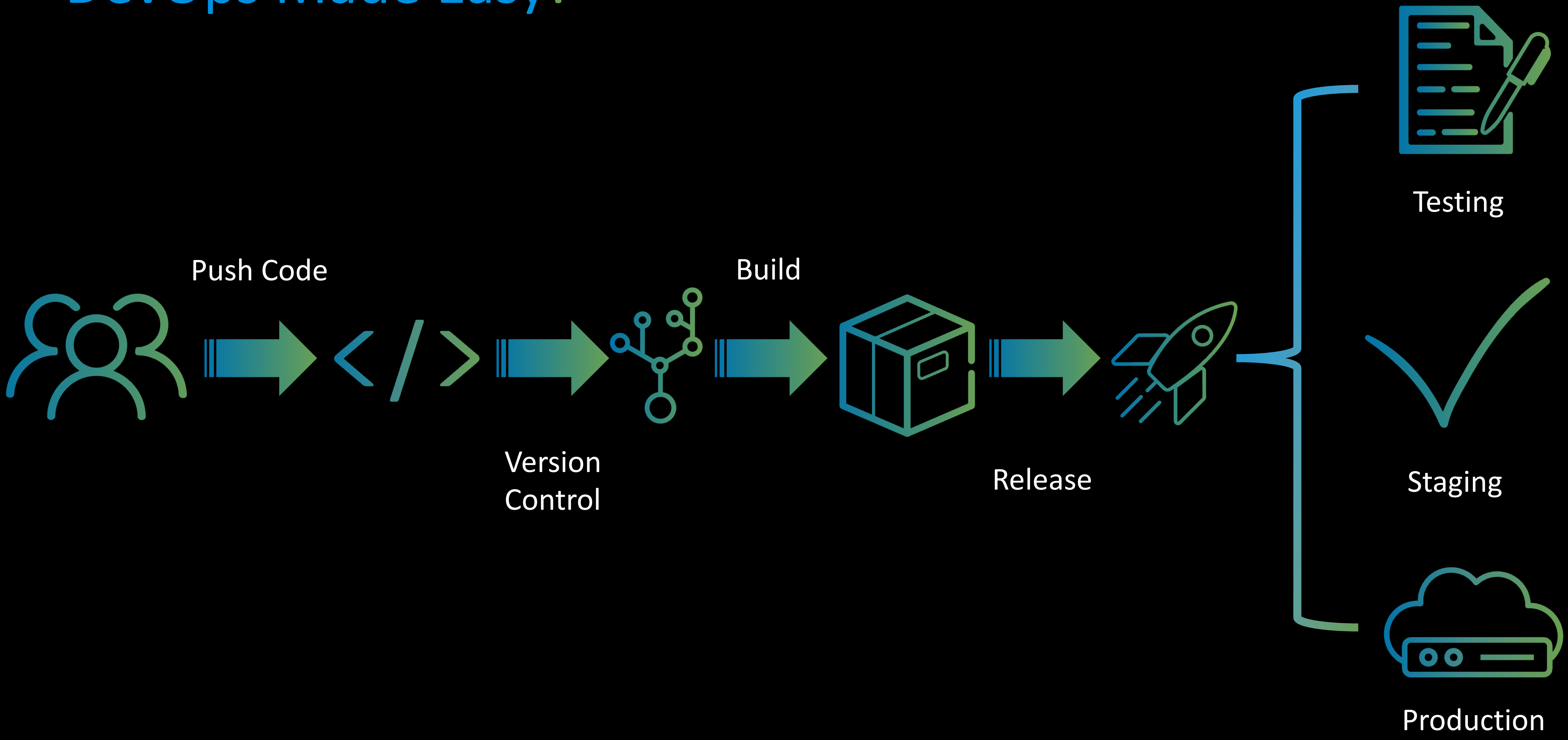
DevOps Made Easy!

As an APEX Developer, you might be looking to apply modern development methodologies and tools used in other development platforms to your Oracle APEX Low-Code Projects.

This includes:

- Git-based code version management
- Code review
- Continuous delivery of apps from one instance to another
- Tracking issues
- Managing your team development

DevOps Made Easy!



Hyand

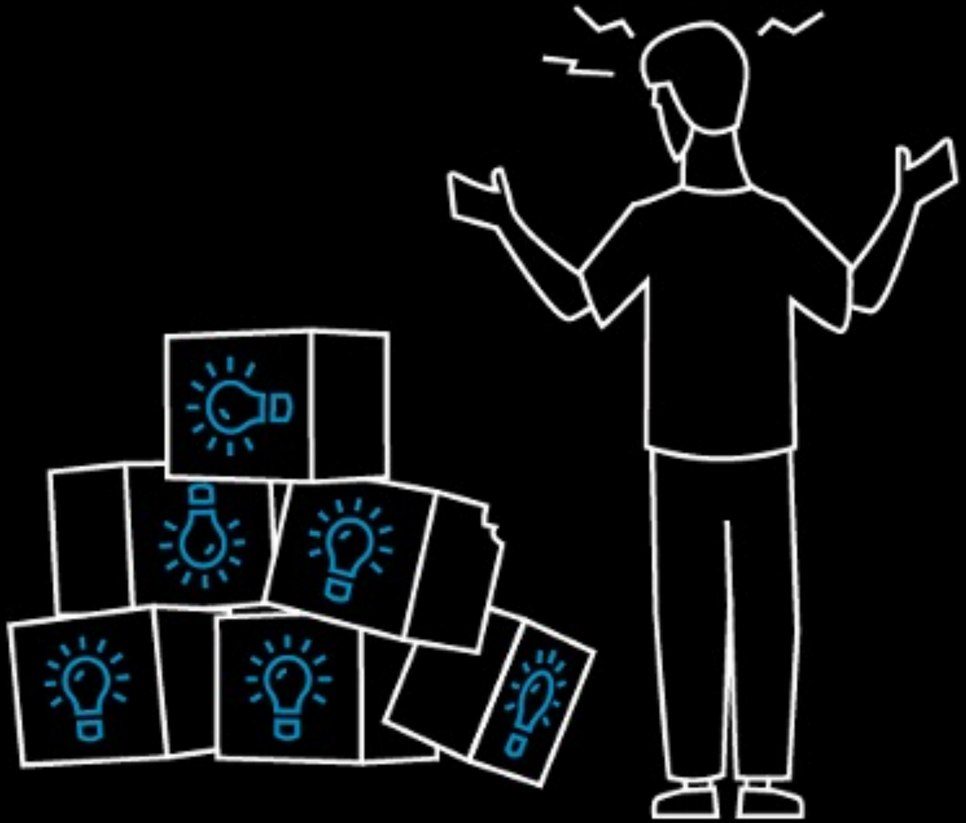
But why Easy?

But why Easy?

The Oracle Autonomous Database (ADB) includes the DBMS_CLOUD_REPO package, an extremely powerful package that provides easy access to files in Cloud Code (Git) Repositories.

With this package, you can:

- Manage repositories
- Handle code in a repository
- Export database schemas and objects
- Execute SQL statements from committed files



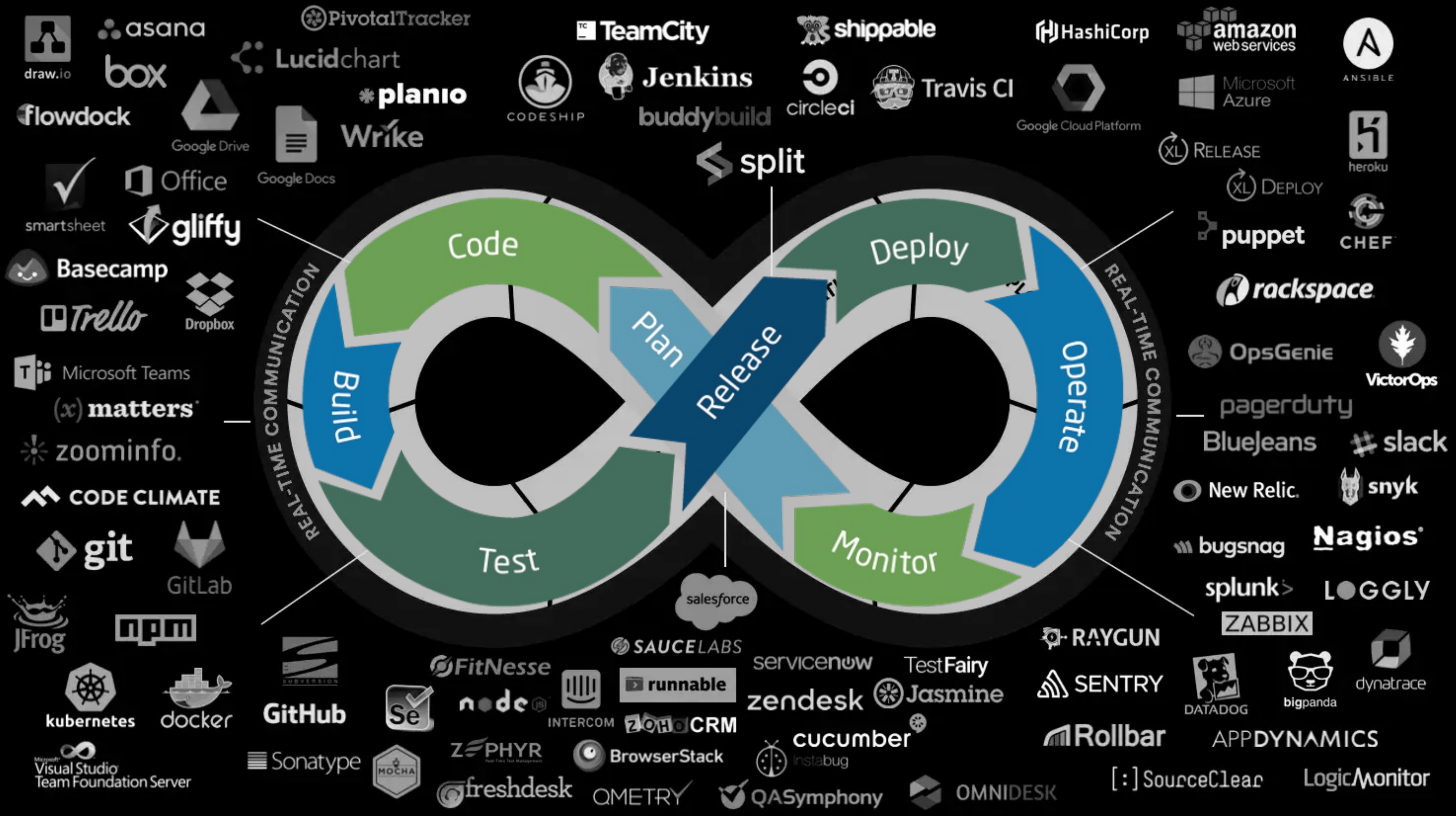
Supported Code Repositories.

GitHub

Azure Repos

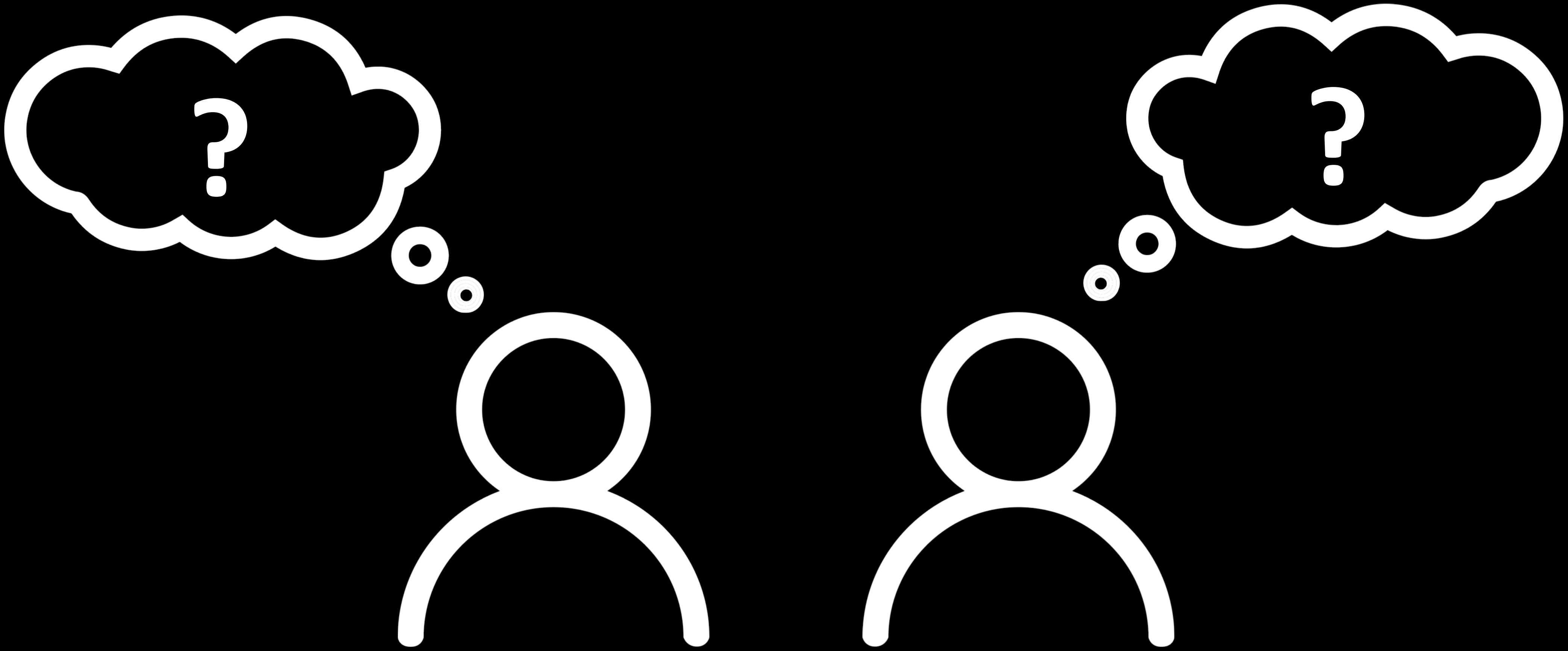
AWS CodeCommit

Wondering why you should use the package instead of relying on proven tools like Jenkins?


Plan
Code
Build
Test
Release
Deploy
Operate
Monitor
REAL-TIME COMMUNICATION
REAL-TIME COMMUNICATION
draw.io
asana
PivotalTracker
box
Lucidchart
planio
flowdock
Google Drive
Google Docs
Wrike
smartsheet
Office
gliffy
Basecamp
Trello
Dropbox
Microsoft Teams
(x) matters
zoominfo.
CODE CLIMATE
git
GitLab
JFrog
npm
kubernetes
docker
GitHub
Visual Studio Team Foundation Server
Sonatype
MOCHA
TeamCity
CODESHIP
Jenkins
buddybuild
shippable
circleci
Travis CI
split
HashiCorp
amazon web services
ANSIBLE
Microsoft Azure
Google Cloud Platform
XL RELEASE
XL DEPLOY
heroku
puppet
CHEF
rackspace
OpsGenie
VictorOps
pagerduty
BlueJeans
slack
New Relic.
snyk
bugsnag
Nagios
splunk>
LOGGLY
ZABBIX
RAYGUN
SENTRY
DATADOG
bigpanda
dynatrace
Rollbar
APPDYNAMICS
[:] SourceClear
LogicMonitor
salesforce
SAUCELABS
FitNesse
node
INTERCOM
runnable
ZOHO CRM
ZEPHYR
BrowserStack
freshdesk
QMETRY
servicenow
TestFairy
zendesk
Jasmine
cucumber
Instabug
QASymphony
OMNIDESK

Hyand

WHASUPP!?



Simple! It's ideal for PL/SQL enthusiasts who are looking for a proven CI/CD strategy for smaller projects with limited budgets and don't have a large team, the necessary expertise or enough time!

Agenda.

1

Introduction

2

Let´s dive deeper

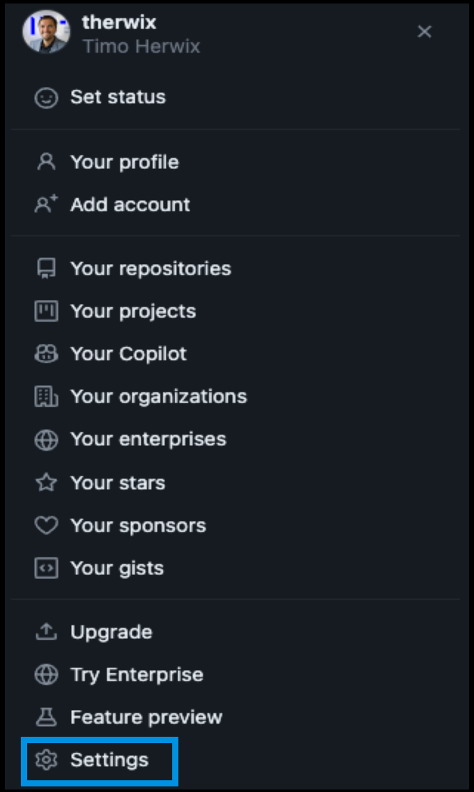
3

Wrap-up

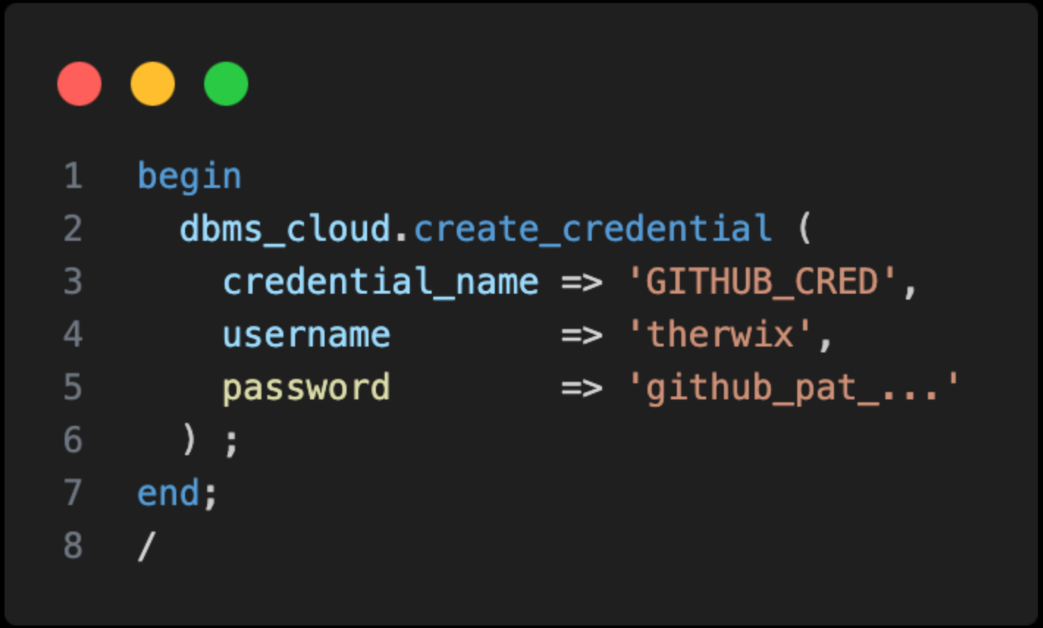
Hyand

Create the credential for interacting with your GitHub Repository.

Create a Personal Access Token



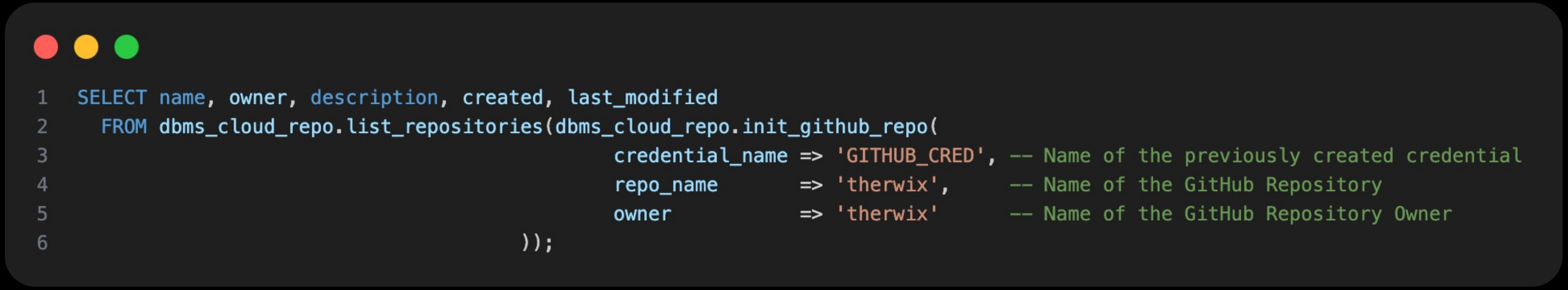
Save your PAT in a Cloud Service Credential



```
begin
  dbms_cloud.create_credential (
    credential_name => 'GITHUB_CRED',
    username        => 'therwix',
    password        => 'github_pat_...'
  ) ;
end;
/
```

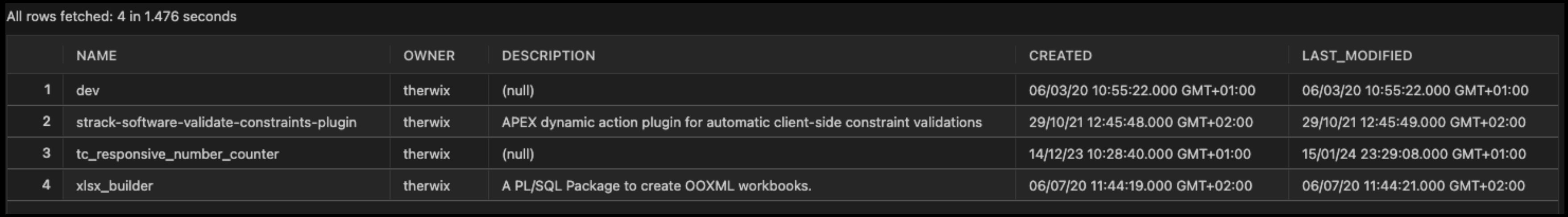
Hyand

Check if the access works.



```sql
SELECT name, owner, description, created, last_modified
  FROM dbms_cloud_repo.list_repositories(dbms_cloud_repo.init_github_repo(
                                             credential_name => 'GITHUB_CRED', -- Name of the previously created credential
                                             repo_name       => 'therwix',      -- Name of the GitHub Repository
                                             owner           => 'therwix'       -- Name of the GitHub Repository Owner
                                     ));
```

If everything works well with the credential setup,
you should see a list of repositories that you can access.

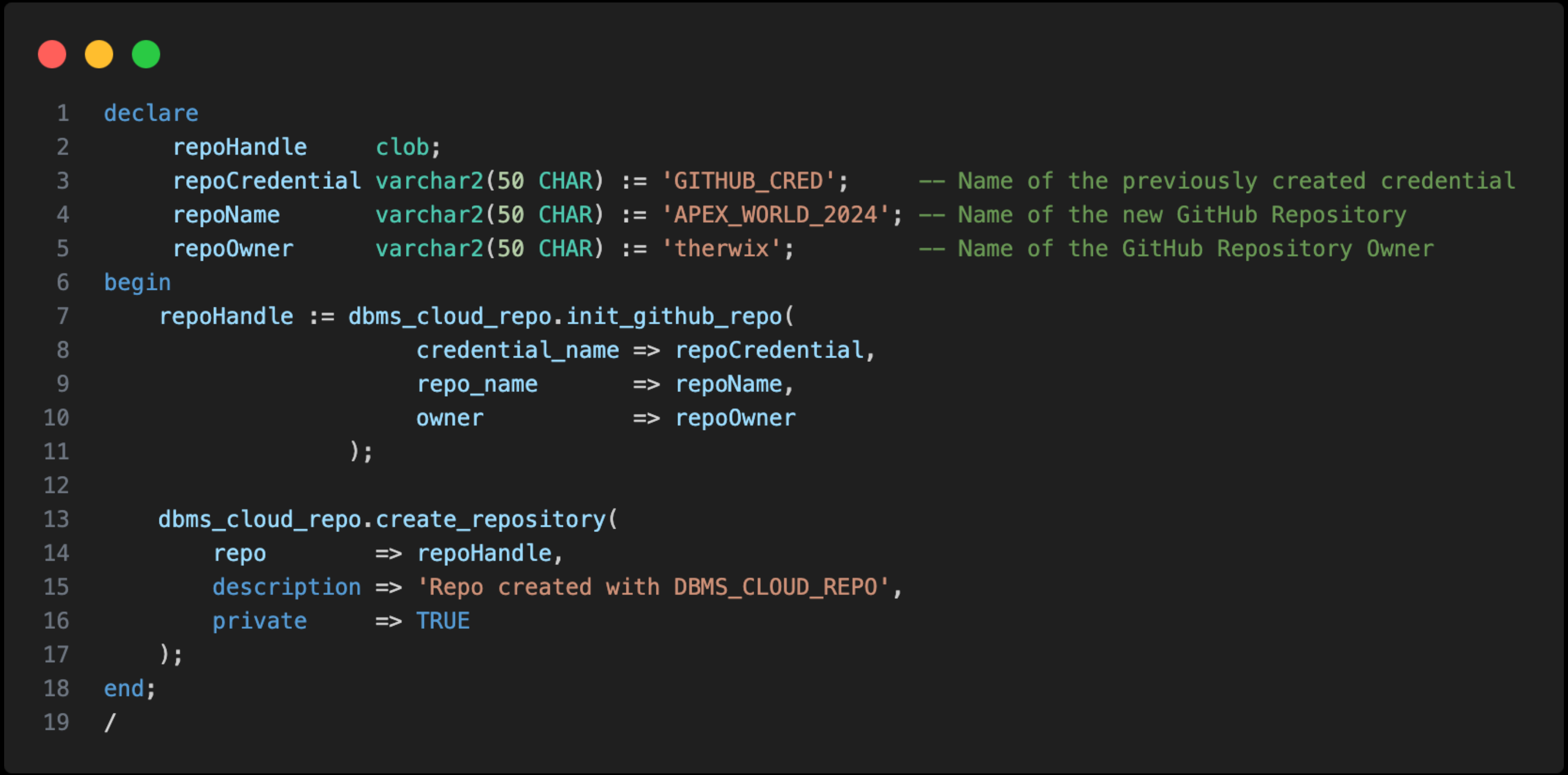


All rows fetched: 4 in 1.476 seconds

	NAME	OWNER	DESCRIPTION	CREATED	LAST_MODIFIED
1	dev	therwix	(null)	06/03/20 10:55:22.000 GMT+01:00	06/03/20 10:55:22.000 GMT+01:00
2	strack-software-validate-constraints-plugin	therwix	APEX dynamic action plugin for automatic client-side constraint validations	29/10/21 12:45:48.000 GMT+02:00	29/10/21 12:45:49.000 GMT+02:00
3	tc_responsive_number_counter	therwix	(null)	14/12/23 10:28:40.000 GMT+01:00	15/01/24 23:29:08.000 GMT+01:00
4	xlsx_builder	therwix	A PL/SQL Package to create OOXML workbooks.	06/07/20 11:44:19.000 GMT+02:00	06/07/20 11:44:21.000 GMT+02:00

Get started with Repository interaction!

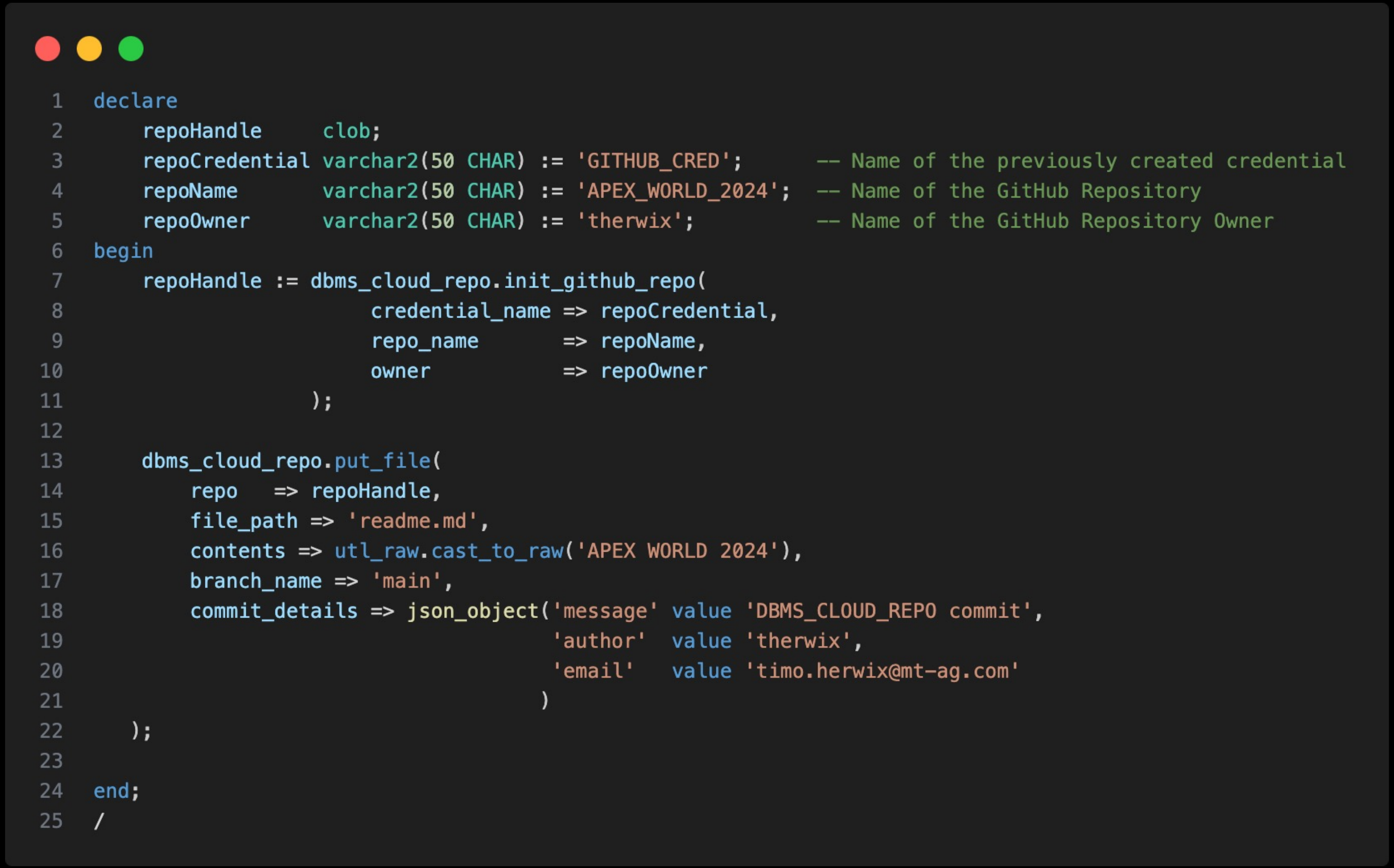
Create a new Repository.



```
declare
     repoHandle     clob;
     repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
     repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024'; -- Name of the new GitHub Repository
     repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.create_repository(
        repo        => repoHandle,
        description => 'Repo created with DBMS_CLOUD_REPO',
        private     => TRUE
    );
end;
/
```

Hyand

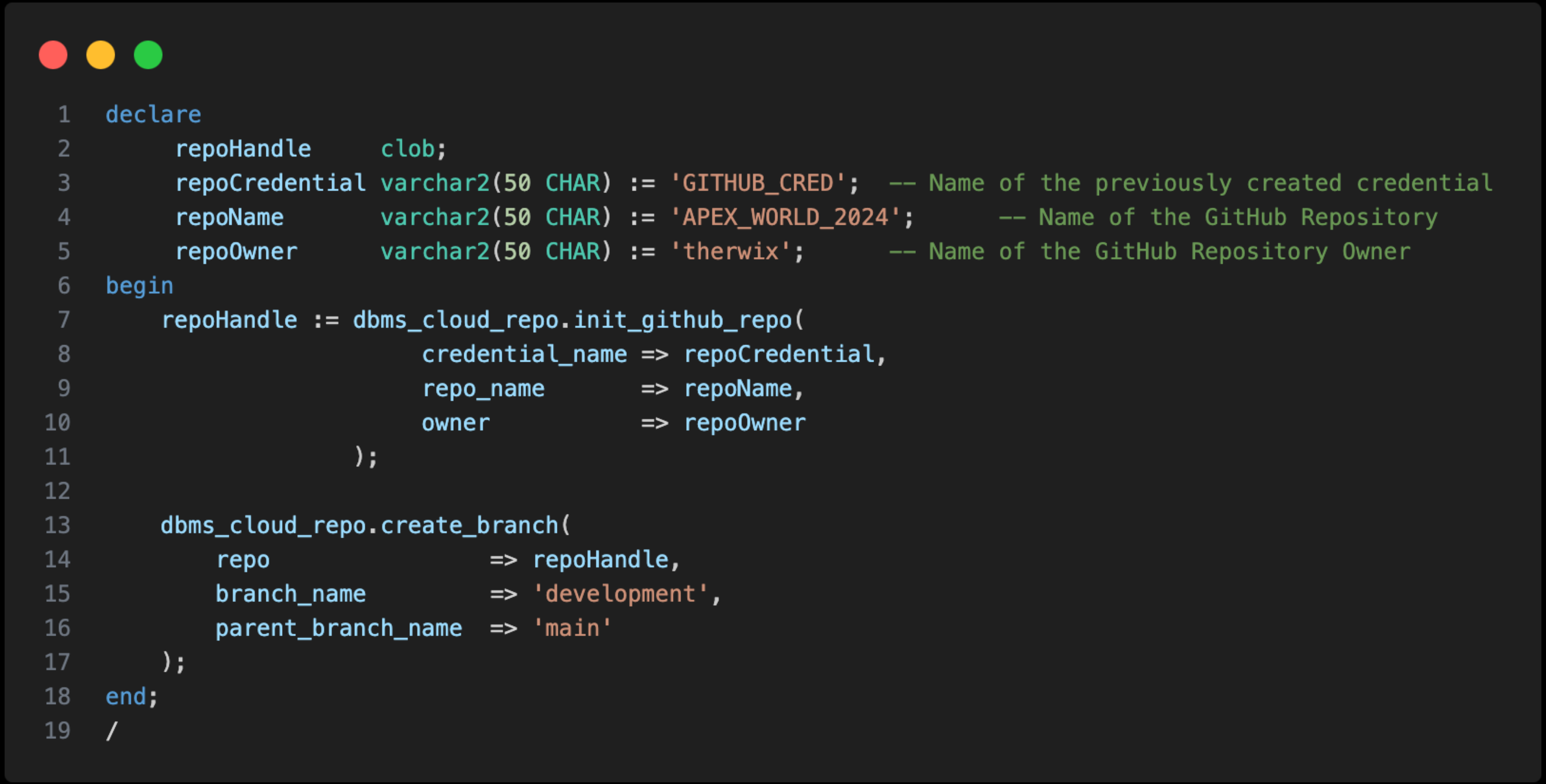
Initialize a new Repository.



```
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.put_file(
        repo    => repoHandle,
        file_path => 'readme.md',
        contents => utl_raw.cast_to_raw('APEX WORLD 2024'),
        branch_name => 'main',
        commit_details => json_object('message' value 'DBMS_CLOUD_REPO commit',
                                      'author'  value 'therwix',
                                      'email'   value 'timo.herwix@mt-ag.com'
                                     )
   );

end;
/
```

Hyand

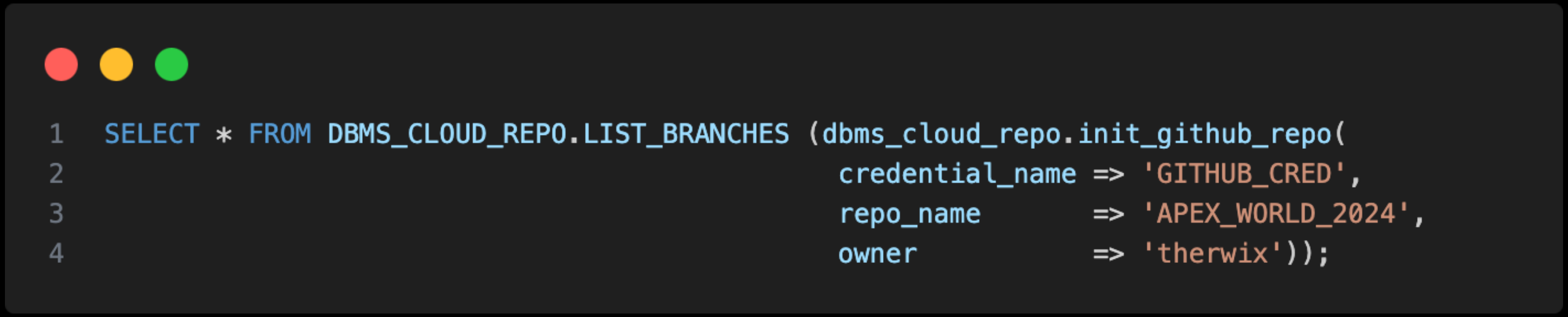
Create a new Branch.



```
declare
     repoHandle     clob;
     repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';  -- Name of the previously created credential
     repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';      -- Name of the GitHub Repository
     repoOwner      varchar2(50 CHAR) := 'therwix';      -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.create_branch(
        repo                => repoHandle,
        branch_name         => 'development',
        parent_branch_name  => 'main'
    );
end;
/
```

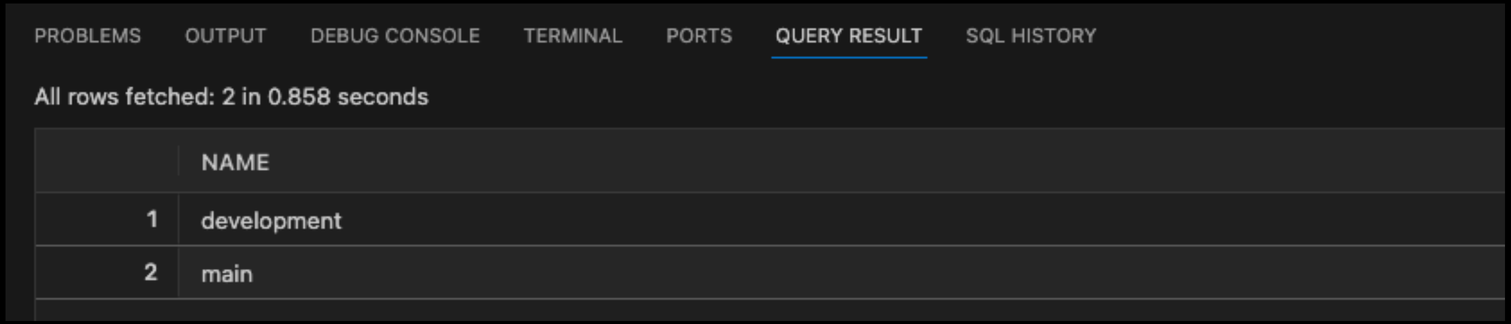
Hyand

Display all branches in a repository.



```
SELECT * FROM DBMS_CLOUD_REPO.LIST_BRANCHES (dbms_cloud_repo.init_github_repo(
                                              credential_name => 'GITHUB_CRED',
                                              repo_name       => 'APEX_WORLD_2024',
                                              owner           => 'therwix'));
```

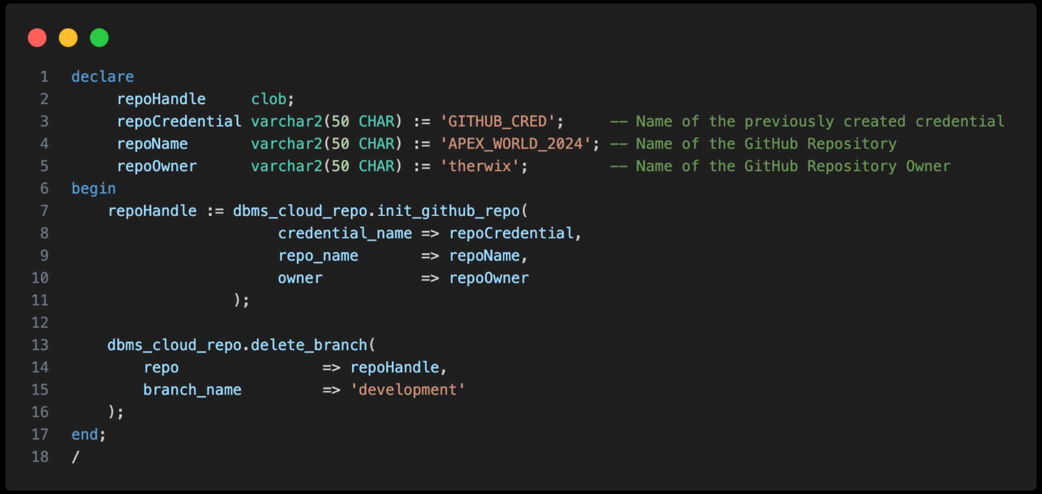
If everything works well,
you should see a list with the branches of your Repository.



```
All rows fetched: 2 in 0.858 seconds

     NAME
  1  development
  2  main
```

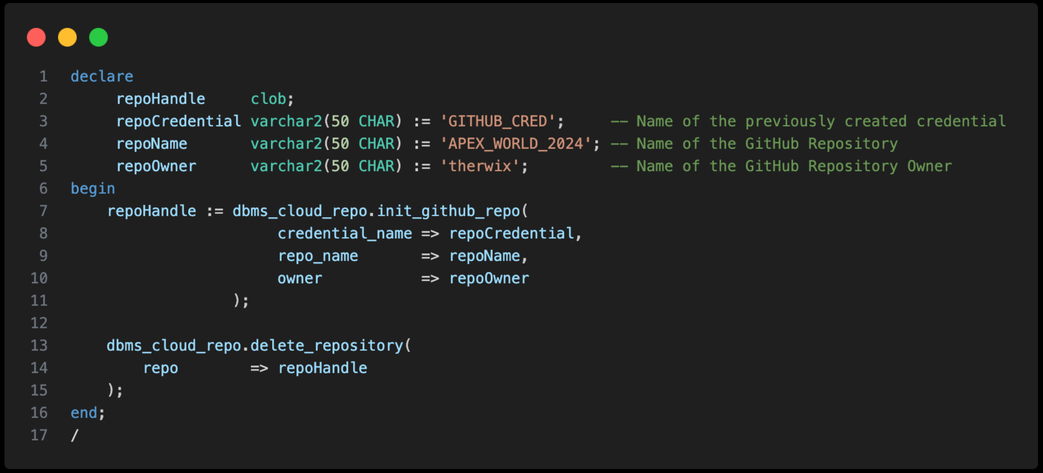
Clean-up!

Delete a Branch



```
declare
     repoHandle     clob;
     repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';     -- Name of the previously created credential
     repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024'; -- Name of the GitHub Repository
     repoOwner      varchar2(50 CHAR) := 'therwix';         -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.delete_branch(
        repo                => repoHandle,
        branch_name         => 'development'
    );
end;
/
```

Delete a Repository



```
declare
     repoHandle     clob;
     repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';     -- Name of the previously created credential
     repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024'; -- Name of the GitHub Repository
     repoOwner      varchar2(50 CHAR) := 'therwix';         -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.delete_repository(
        repo        => repoHandle
    );
end;
/
```

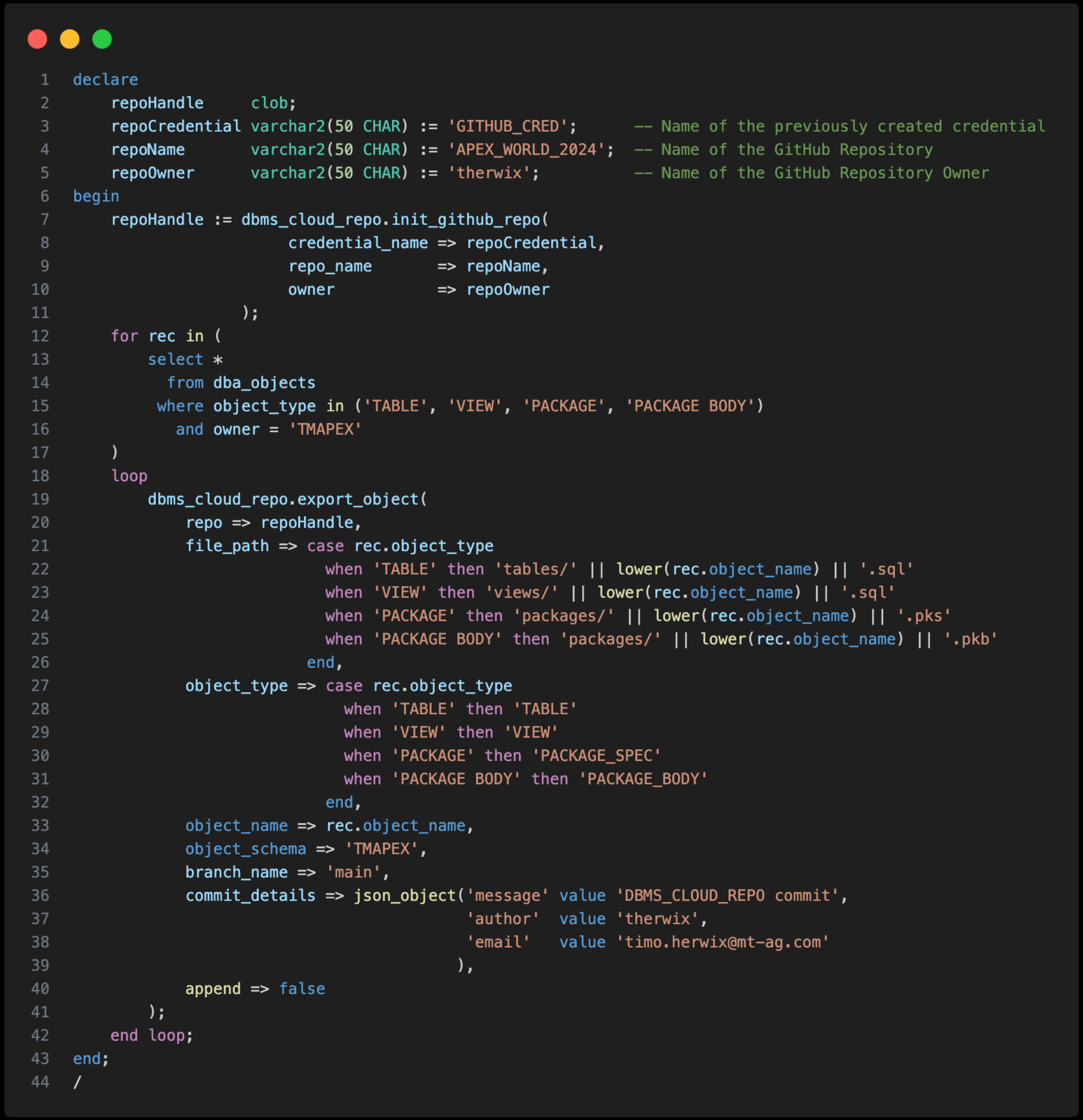
Hyand

Hyand

Moving content to the Code Repository!

Hyand

Export all schema objects to single files.

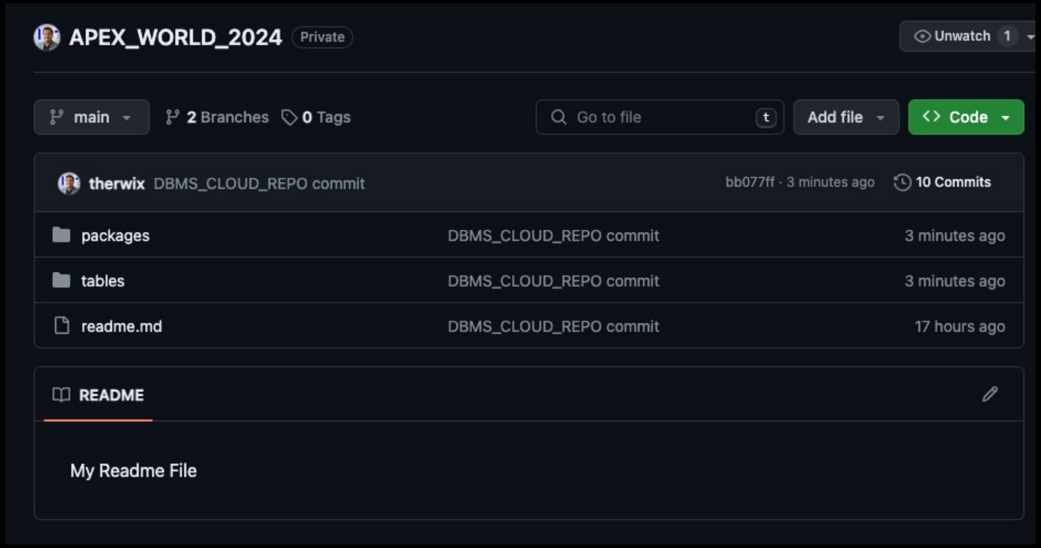


```sql
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );
    for rec in (
        select *
          from dba_objects
         where object_type in ('TABLE', 'VIEW', 'PACKAGE', 'PACKAGE BODY')
           and owner = 'TMAPEX'
    )
    loop
        dbms_cloud_repo.export_object(
            repo => repoHandle,
            file_path => case rec.object_type
                           when 'TABLE' then 'tables/' || lower(rec.object_name) || '.sql'
                           when 'VIEW' then 'views/' || lower(rec.object_name) || '.sql'
                           when 'PACKAGE' then 'packages/' || lower(rec.object_name) || '.pks'
                           when 'PACKAGE BODY' then 'packages/' || lower(rec.object_name) || '.pkb'
                         end,
            object_type => case rec.object_type
                             when 'TABLE' then 'TABLE'
                             when 'VIEW' then 'VIEW'
                             when 'PACKAGE' then 'PACKAGE_SPEC'
                             when 'PACKAGE BODY' then 'PACKAGE_BODY'
                           end,
            object_name => rec.object_name,
            object_schema => 'TMAPEX',
            branch_name => 'main',
            commit_details => json_object('message' value 'DBMS_CLOUD_REPO commit',
                                          'author'  value 'therwix',
                                          'email'   value 'timo.herwix@mt-ag.com'
                                         ),
            append => false
        );
    end loop;
end;
/
```

Hyand

Export all schema objects to single files.

View of the Repository



APEX_WORLD_2024 Private

Unwatch 1

main · 2 Branches · 0 Tags · Go to file · Add file · Code

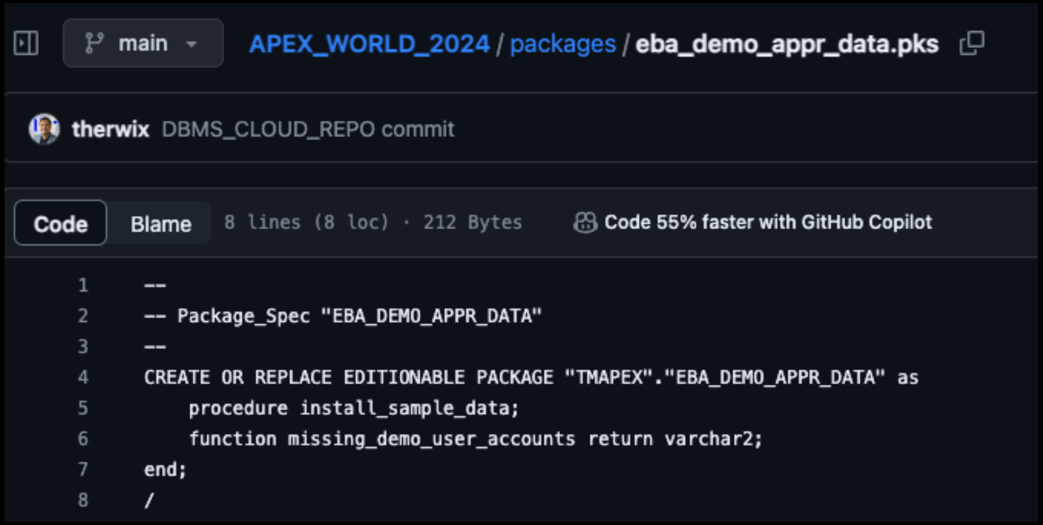
therwix DBMS_CLOUD_REPO commit · bb077ff · 3 minutes ago · 10 Commits

packages	DBMS_CLOUD_REPO commit	3 minutes ago
tables	DBMS_CLOUD_REPO commit	3 minutes ago
readme.md	DBMS_CLOUD_REPO commit	17 hours ago

README

My Readme File

View of a DDL-Script



main · APEX_WORLD_2024 / packages / eba_demo_appr_data.pks

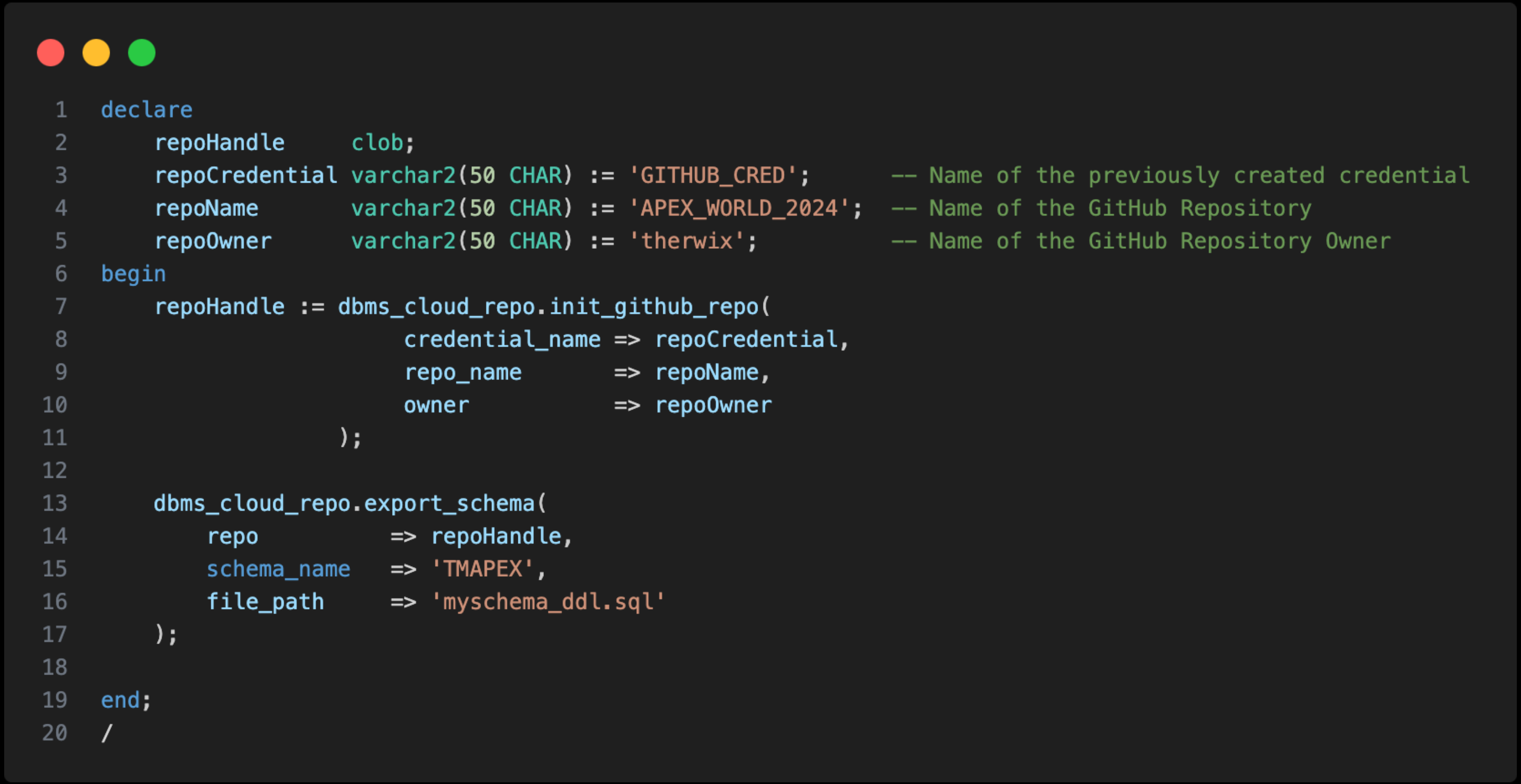
therwix DBMS_CLOUD_REPO commit

Code · Blame · 8 lines (8 loc) · 212 Bytes · Code 55% faster with GitHub Copilot

```
--
-- Package_Spec "EBA_DEMO_APPR_DATA"
--
CREATE OR REPLACE EDITIONABLE PACKAGE "TMAPEX"."EBA_DEMO_APPR_DATA" as
    procedure install_sample_data;
    function missing_demo_user_accounts return varchar2;
end;
/
```

Hyand

Export all schema objects to a single file.



```
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                        credential_name => repoCredential,
                        repo_name       => repoName,
                        owner           => repoOwner
                    );

    dbms_cloud_repo.export_schema(
        repo          => repoHandle,
        schema_name   => 'TMAPEX',
        file_path     => 'myschema_ddl.sql'
    );

end;
/
```

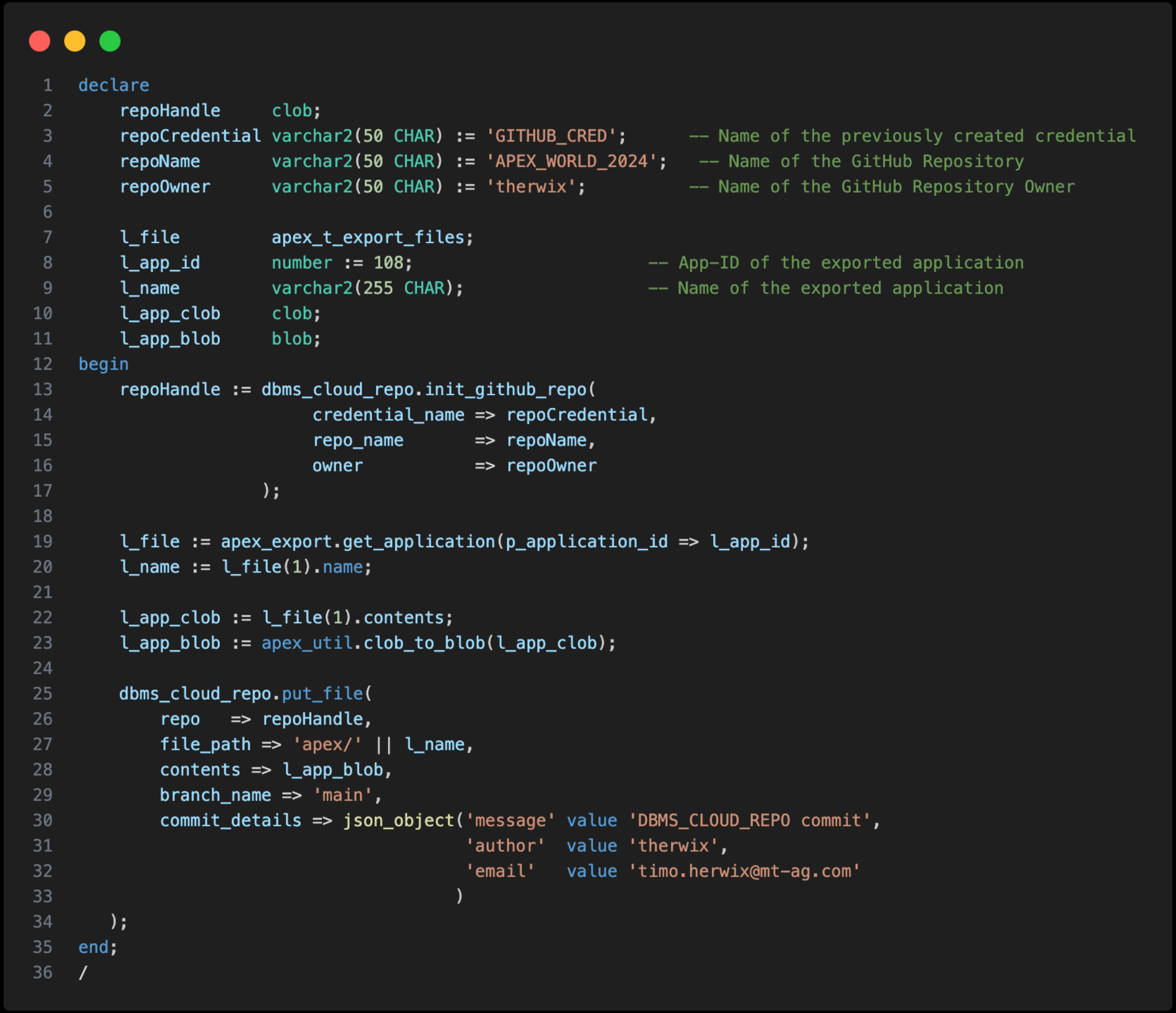
Hyand

Exporting an APEX Application!

You can easily export APEX applications to a Code repository too.
All you need to do is call the APEX_EXPORT package and pass the application ID to the GET_APPLICATION function.

But, there's a tiny thing to remember: the output of GET_APPLICATION, which is a CLOB, needs to be converted to a BLOB to work with the DBMS_CLOUD_REPO.PUT_FILE procedure.
However, APEX_UTIL has a helpful function to do this.

Exporting an APEX Application.



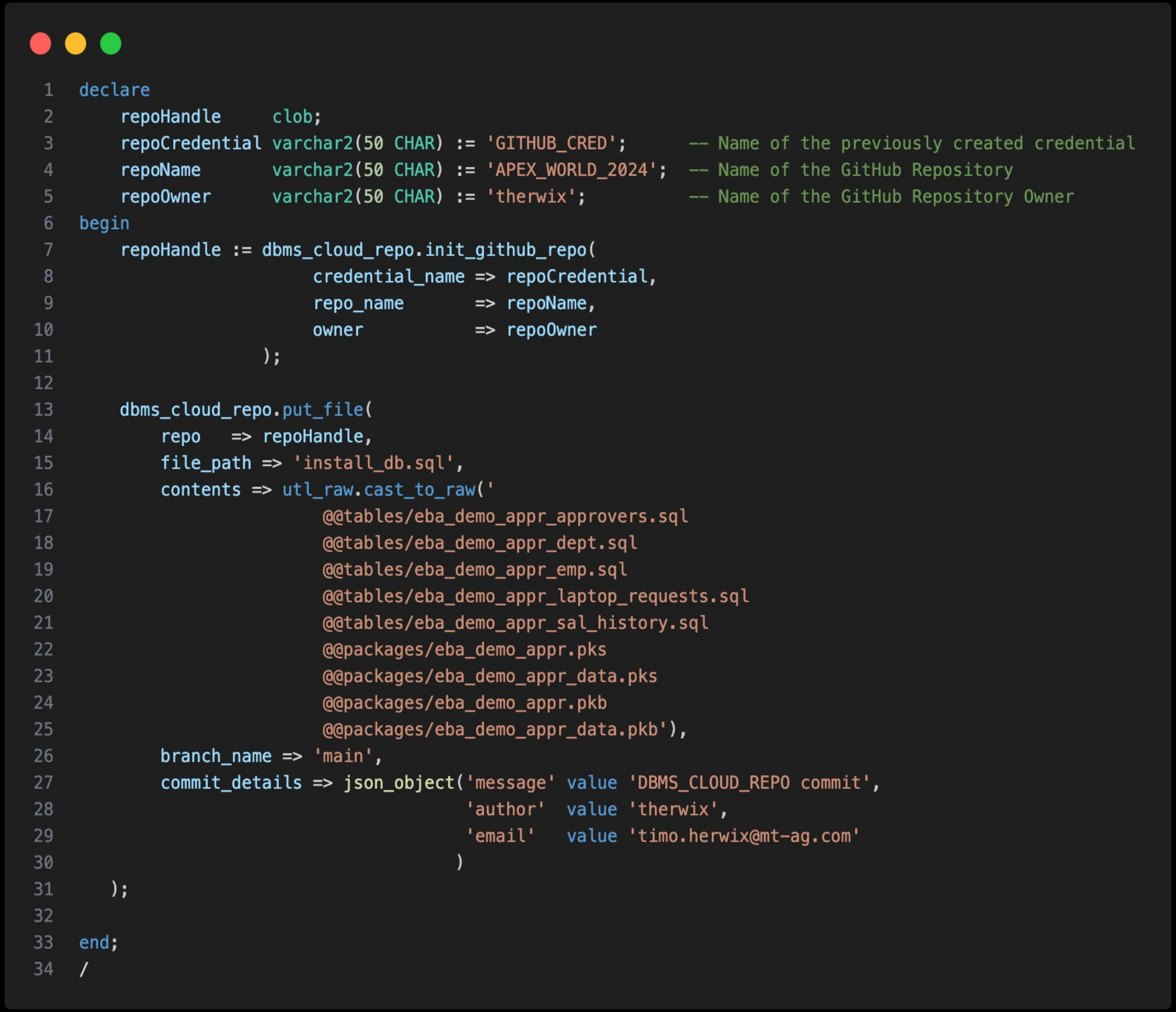
```
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';   -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner

    l_file         apex_t_export_files;
    l_app_id       number := 108;                       -- App-ID of the exported application
    l_name         varchar2(255 CHAR);                  -- Name of the exported application
    l_app_clob     clob;
    l_app_blob     blob;
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    l_file := apex_export.get_application(p_application_id => l_app_id);
    l_name := l_file(1).name;

    l_app_clob := l_file(1).contents;
    l_app_blob := apex_util.clob_to_blob(l_app_clob);

    dbms_cloud_repo.put_file(
        repo   => repoHandle,
        file_path => 'apex/' || l_name,
        contents => l_app_blob,
        branch_name => 'main',
        commit_details => json_object('message' value 'DBMS_CLOUD_REPO commit',
                                      'author'  value 'therwix',
                                      'email'   value 'timo.herwix@mt-ag.com'
                                     )
   );
end;
/
```

Hyand

Hyland

Perform SQL Operations from Code Repositories!

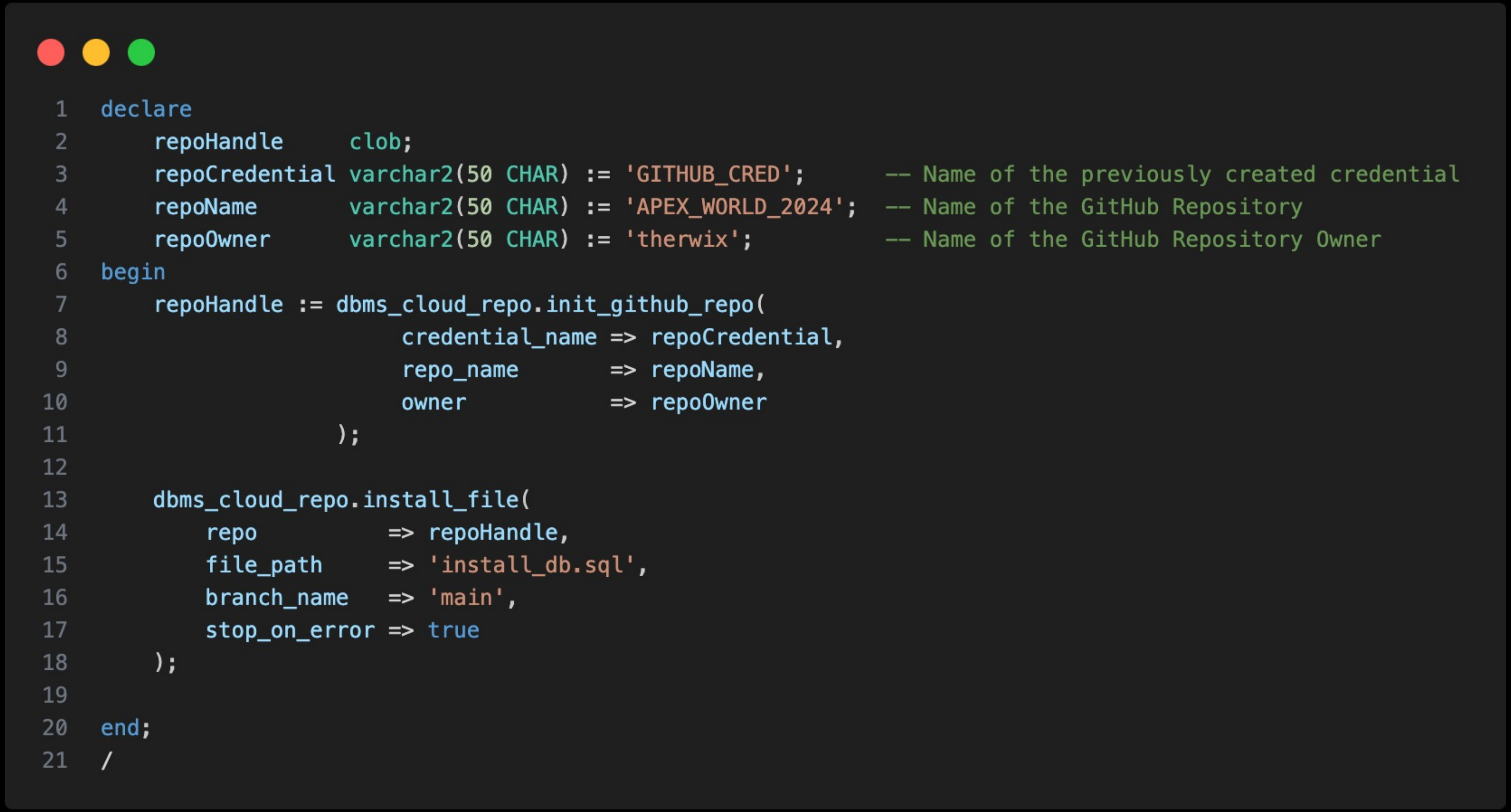
Creating an installation script.



```sql
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.put_file(
        repo    => repoHandle,
        file_path => 'install_db.sql',
        contents => utl_raw.cast_to_raw('
                        @@tables/eba_demo_appr_approvers.sql
                        @@tables/eba_demo_appr_dept.sql
                        @@tables/eba_demo_appr_emp.sql
                        @@tables/eba_demo_appr_laptop_requests.sql
                        @@tables/eba_demo_appr_sal_history.sql
                        @@packages/eba_demo_appr.pks
                        @@packages/eba_demo_appr_data.pks
                        @@packages/eba_demo_appr.pkb
                        @@packages/eba_demo_appr_data.pkb'),
        branch_name => 'main',
        commit_details => json_object('message' value 'DBMS_CLOUD_REPO commit',
                                      'author'  value 'therwix',
                                      'email'   value 'timo.herwix@mt-ag.com'
                                     )
   );

end;
/
```

Hyand

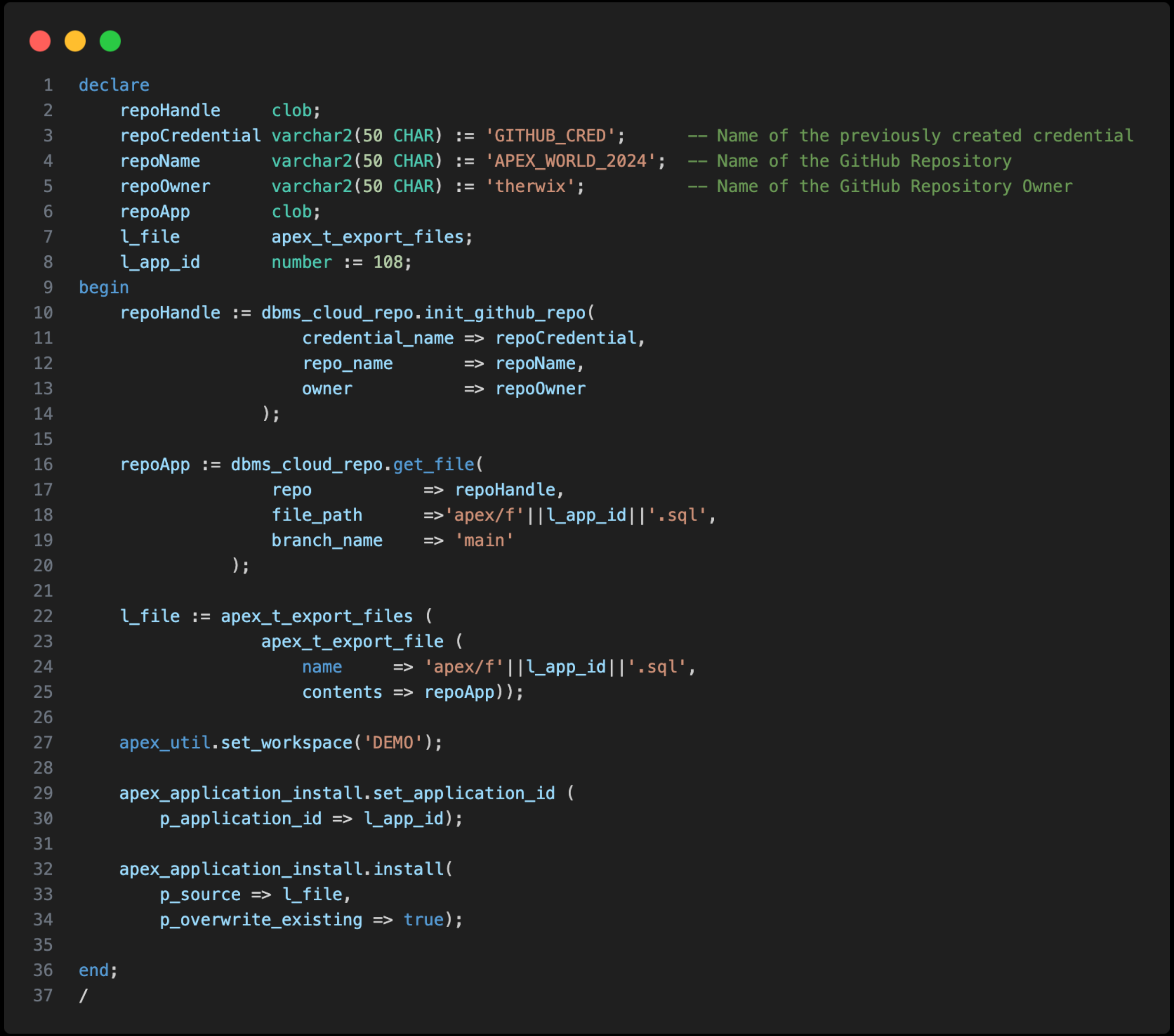
Execute the installation script.



```
declare
    repoHandle      clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                      credential_name => repoCredential,
                      repo_name       => repoName,
                      owner           => repoOwner
                 );

    dbms_cloud_repo.install_file(
        repo          => repoHandle,
        file_path     => 'install_db.sql',
        branch_name   => 'main',
        stop_on_error => true
    );

end;
/
```

Hyand

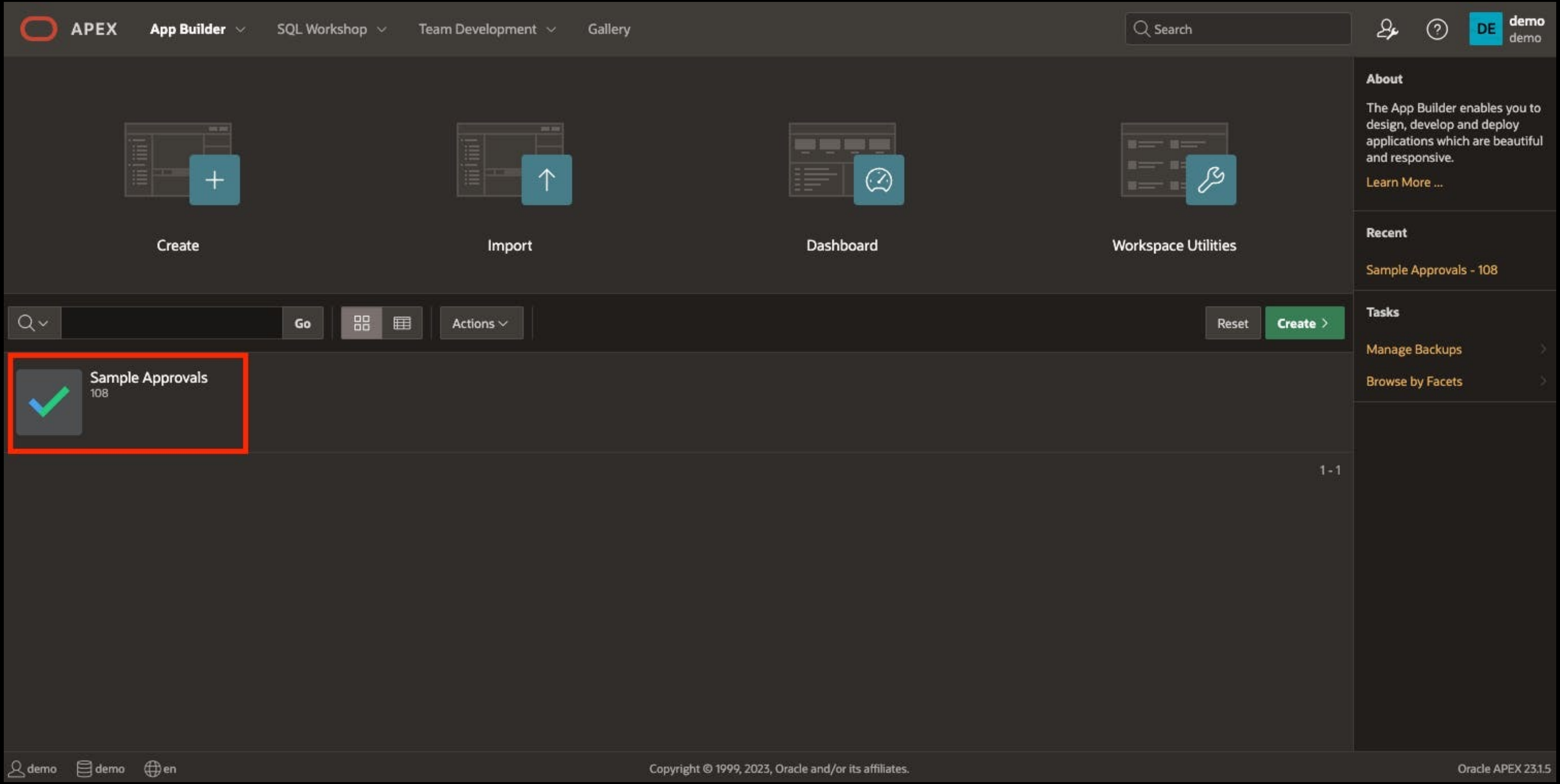
Deploy an APEX Application from a Script.



```
declare
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
    repoApp        clob;
    l_file         apex_t_export_files;
    l_app_id       number := 108;
begin
    repoHandle := dbms_cloud_repo.init_github_repo(
                      credential_name => repoCredential,
                      repo_name       => repoName,
                      owner           => repoOwner
                  );

    repoApp := dbms_cloud_repo.get_file(
                   repo           => repoHandle,
                   file_path      =>'apex/f'||l_app_id||'.sql',
                   branch_name    => 'main'
               );

    l_file := apex_t_export_files (
                  apex_t_export_file (
                      name     => 'apex/f'||l_app_id||'.sql',
                      contents => repoApp));

    apex_util.set_workspace('DEMO');

    apex_application_install.set_application_id (
        p_application_id => l_app_id);

    apex_application_install.install(
        p_source => l_file,
        p_overwrite_existing => true);

end;
/
```

Hyand

Deploy an APEX Application from a Script.



About

The App Builder enables you to design, develop and deploy applications which are beautiful and responsive.

Learn More ...

Recent

Sample Approvals - 108

Tasks

Manage Backups

Browse by Facets

Copyright © 1999, 2023, Oracle and/or its affiliates.

Oracle APEX 23.1.5

Hyland

What is about the table differences between our environments???

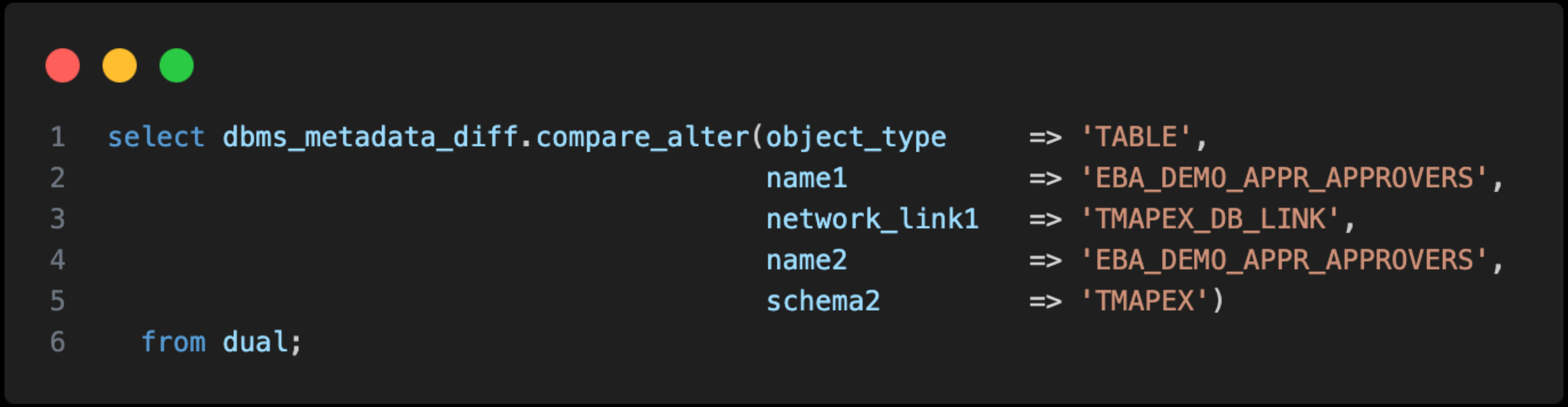
Get started with Liquibase!

Liquibase

...or you are a PL/SQL nerd and use DBMS_METADATA_DIFF 🤓

This solution works well if you make Database Links between your Autonomous instances.

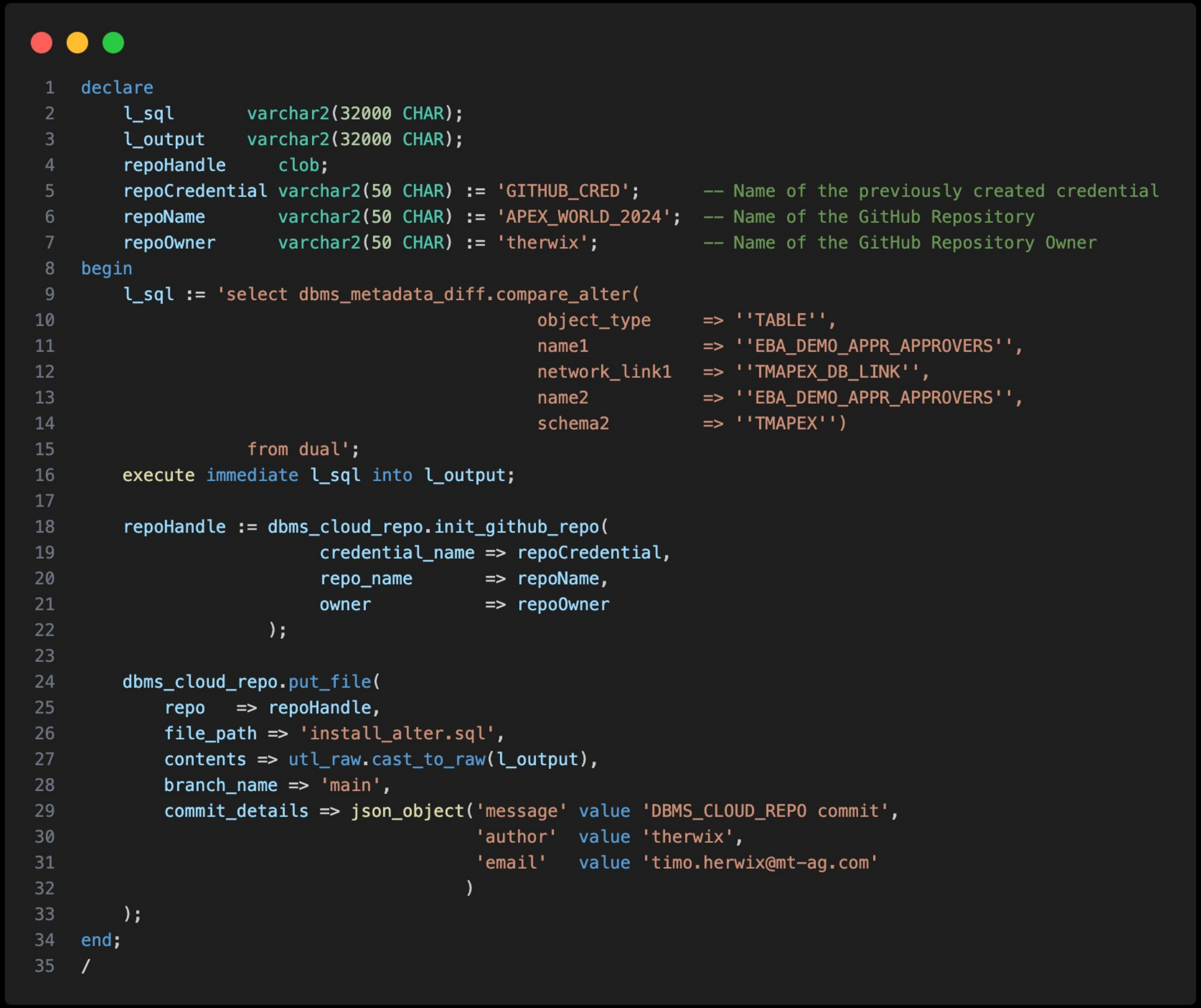
Take a look at the differences in our tables and push them.



```sql
select dbms_metadata_diff.compare_alter(object_type     => 'TABLE',
                                        name1           => 'EBA_DEMO_APPR_APPROVERS',
                                        network_link1   => 'TMAPEX_DB_LINK',
                                        name2           => 'EBA_DEMO_APPR_APPROVERS',
                                        schema2         => 'TMAPEX')
  from dual;
```

ALTER TABLE "TMAPEX"."EBA_DEMO_APPR_APPROVERS"
ADD ("MAX_SALARY" NUMBER);

Take a look at the differences in our tables and push them.



```sql
declare
    l_sql       varchar2(32000 CHAR);
    l_output    varchar2(32000 CHAR);
    repoHandle     clob;
    repoCredential varchar2(50 CHAR) := 'GITHUB_CRED';      -- Name of the previously created credential
    repoName       varchar2(50 CHAR) := 'APEX_WORLD_2024';  -- Name of the GitHub Repository
    repoOwner      varchar2(50 CHAR) := 'therwix';          -- Name of the GitHub Repository Owner
begin
    l_sql := 'select dbms_metadata_diff.compare_alter(
                                            object_type     => ''TABLE'',
                                            name1           => ''EBA_DEMO_APPR_APPROVERS'',
                                            network_link1   => ''TMAPEX_DB_LINK'',
                                            name2           => ''EBA_DEMO_APPR_APPROVERS'',
                                            schema2         => ''TMAPEX'')
                from dual';
    execute immediate l_sql into l_output;

    repoHandle := dbms_cloud_repo.init_github_repo(
                       credential_name => repoCredential,
                       repo_name       => repoName,
                       owner           => repoOwner
                  );

    dbms_cloud_repo.put_file(
        repo   => repoHandle,
        file_path => 'install_alter.sql',
        contents => utl_raw.cast_to_raw(l_output),
        branch_name => 'main',
        commit_details => json_object('message' value 'DBMS_CLOUD_REPO commit',
                                      'author'  value 'therwix',
                                      'email'   value 'timo.herwix@mt-ag.com'
                                     )
    );
end;
/
```

Hyand

Hyand

Agenda.

1

Introduction

2

Let´s dive deeper

3

Wrap-up

Hyand

Wrap-up.

In conclusion, the DBMS_CLOUD_REPO package simplifies CI/CD workflows for APEX applications on Autonomous Database by providing a single interface for managing Code Repositories.

- Managing branches and repositories
- Exporting database schemas
- Executing SQL scripts directly

Wrap-up.

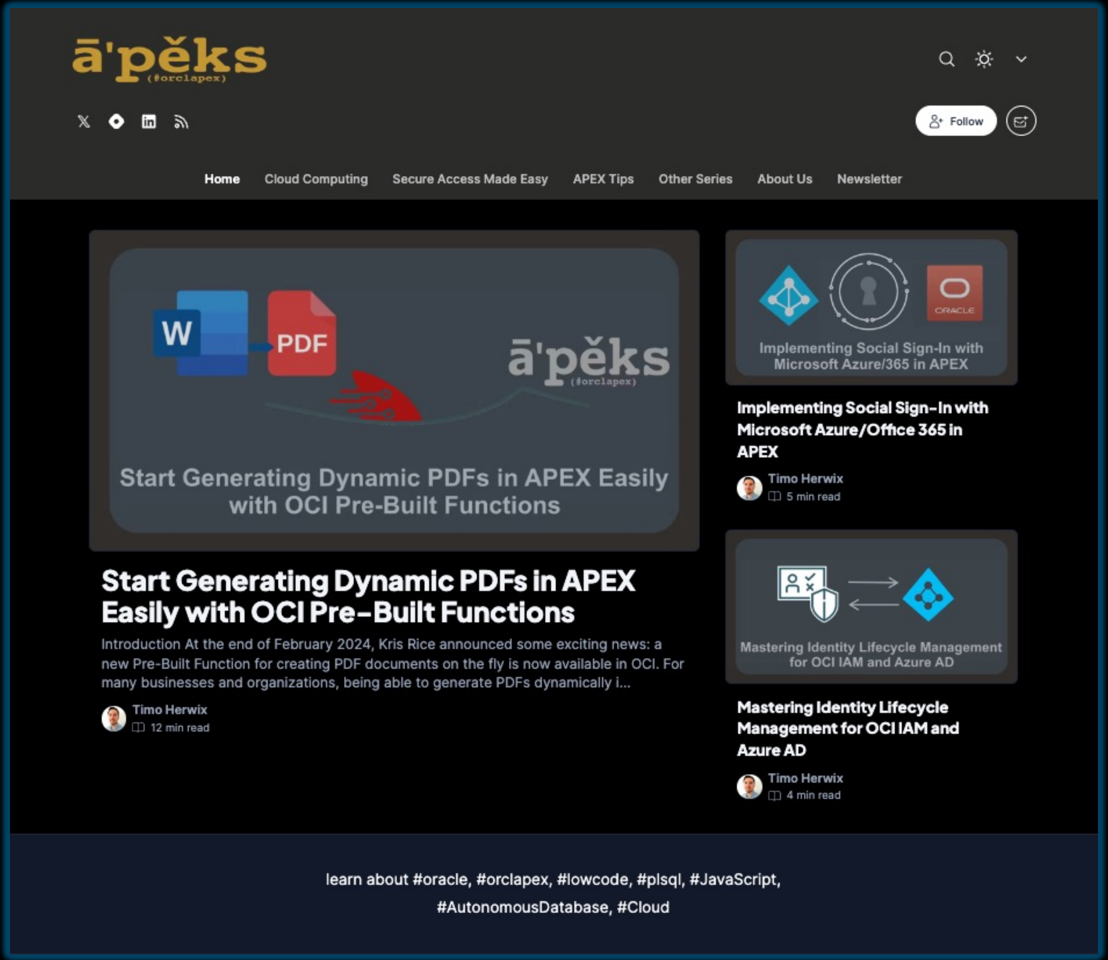
However, it is only intended as a simple solution and cannot replace more complex procedures.

It's perfect for:

- Smaller projects
- Limited budgets
- When expertise is limited
- When time is tight

Blog.

Scan me!


ā'pěks (#orclapex)

Follow

Home · Cloud Computing · Secure Access Made Easy · APEX Tips · Other Series · About Us · Newsletter

W
PDF
ā'pěks (#orclapex)
Start Generating Dynamic PDFs in APEX Easily with OCI Pre-Built Functions

Start Generating Dynamic PDFs in APEX Easily with OCI Pre-Built Functions

Introduction At the end of February 2024, Kris Rice announced some exciting news: a new Pre-Built Function for creating PDF documents on the fly is now available in OCI. For many businesses and organizations, being able to generate PDFs dynamically i...

Timo Herwix
12 min read

ORACLE
Implementing Social Sign-In with Microsoft Azure/365 in APEX

Implementing Social Sign-In with Microsoft Azure/Office 365 in APEX

Timo Herwix
5 min read

Mastering Identity Lifecycle Management for OCI IAM and Azure AD

Timo Herwix
4 min read

learn about #oracle, #orclapex, #lowcode, #plsql, #JavaScript, #AutonomousDatabase, #Cloud

Are you interested?

Timo Herwix
Senior Consultant

Telefon: +49 2102 30 961-0
Mobil: +49 176 20185455
Mail: timo.herwix@hyand.com

Timo Herwix

@Therwix

tm-apex.hashnode.dev

Hyand