

Diese PDF-Version meiner Folien verzichtet auf Hintergrundbilder, weil sonst die Dateigröße das Limit der DOAG von 10 MB überschreitet. Die vollständige HTML-Version steht dauerhaft online unter folgendem Link zur Verfügung: ogobrecht.github.io/posts/2019-11-19-doag-konferenz

SCHNELLSTART

VERSIONSKONTROLLE FÜR EXISTIERENDE ORACLE PROJEKTE

Ottmar Gobrecht [@ogobrecht](#)

DOAG Konferenz Nürnberg
19. November 2019

ZU MEINER PERSON

- Oracle APEX Entwickler seit 2008 (APEX 3.1)
- Seit 2013 im Headquarter der Linde AG
- Individualsoftware für Fachbereiche
- Aktiv im [Open Source Bereich](#)

MOTIVATION

*„There is no clean (database)
development without Version Control“*

Samuel Nitsche ([Blog Post](#))

DEPLOYMENT PAIN?!?

JUST DO IT



PeterRaganitsch

@PeterRaganitsch



Source code control is just awesome and saved me again. Having a history of all old versions of your code is invaluable!

Can't stress that point enough, get your shit together and use git, or anything else. Just do it.

[#code](#) [#orclAPEX](#) [#oracle](#) [#programming](#)

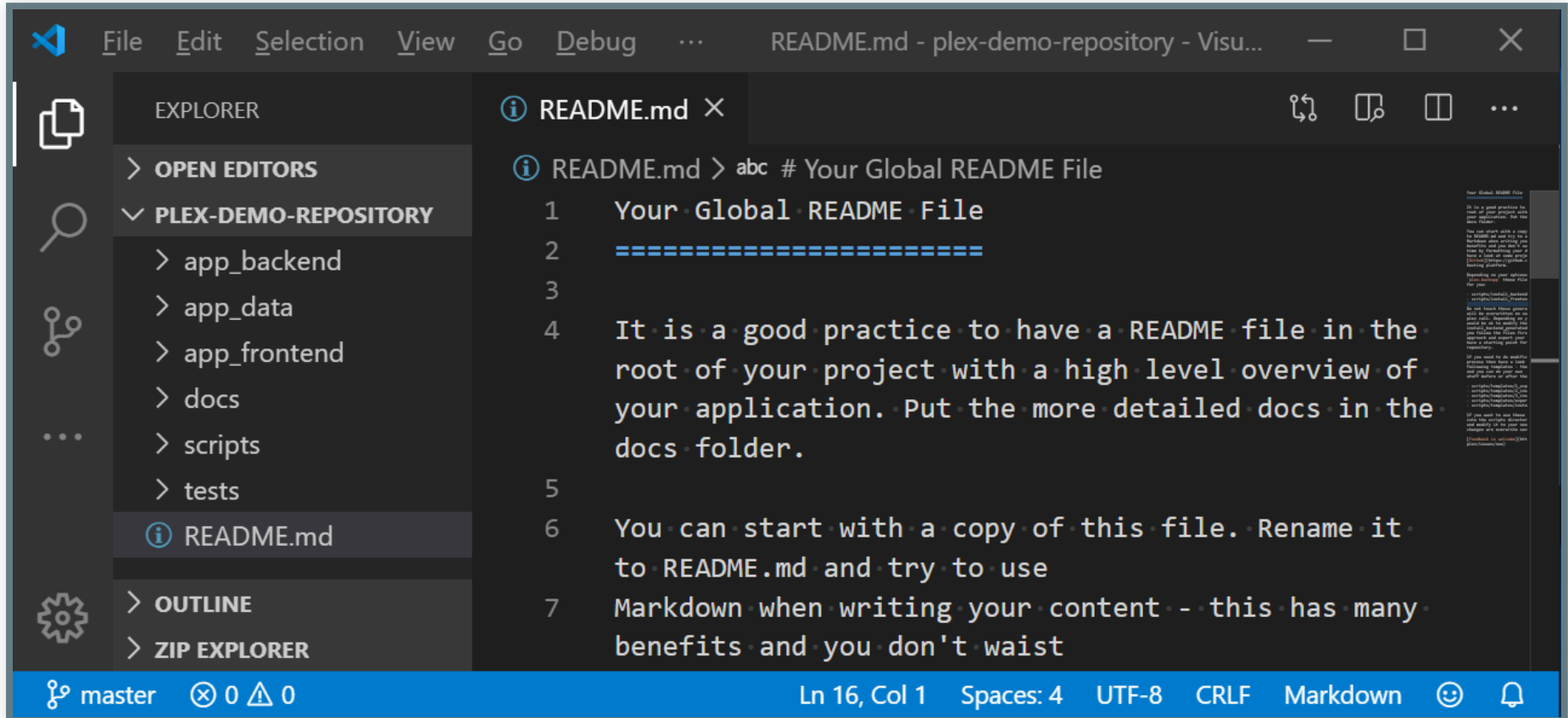
Tweet vom 7. Oktober 2019

UNSER ZIEL

Export „all in one“ für Versionskontrolle

- Backend (Schema DDL)
- Daten (Katalogdaten)
- Frontend (APEX App, zerlegt)
- Deployment Templates
- Wiederanlauffähigkeit
- Übersichtliche Dateistruktur

GEWÜNSCHTE VERZEICHNISSTRUKTUR



Kurze Wege, alle Skripte vereint, übersichtlich

TOOLS

TOOL-VERGLEICH DDL EXPORT - KRITERIEN

Tools: SQL Dev, PL/SQL Dev, Toad, PLEX

- Eine Skript-Datei pro Objekt?
- Unterverzeichnisse pro Objekttyp?
- Eigene Dateien FK Constraints?
- "Object already exist" verhinderbar?
- Daten exportierbar?
- APEX App exportierbar?

TOOL-VERGLEICH DDL EXPORT

Kriterium	SQL Dev.	PL/SQL Dev.	Toad	PLEX
Datei pro Objekt	Ja	Ja	Ja	Ja
Unterverz. pro Typ	Ja	Nein	Ja	Ja
FK Constr. extra	Ja	Nein	Ja	Ja
Verhi. object exist	Nein	Nein	Nein	Ja
Export Daten	Ja	Nein	<i>Jein</i>	Ja
Export APEX App	Ja	Nein	Nein	Ja

ANMERKUNGEN PLEX

- Ist ein Package (**PL**/SQL **Ex**port Utilities)
- Ausgabeverzeichnisstruktur anpassbar
- APEX App zerlegt
(Änderungen im VCS nachvollziehbar)
- [Projekt auf GitHub](#)
- Wenig Nacharbeit erforderlich

ANMERKUNGEN TOAD

- Zwei Exportmöglichkeiten (mindestens)
 - Entweder Unterverzeichnisse pro Objekttyp...
 - ... oder Daten
- Daten nur als Insert Statements
- Umfangreich konfigurierbar, unübersichtlich
- Viel Nacharbeit erforderlich

ANMERKUNGEN PL/SQL DEVELOPER

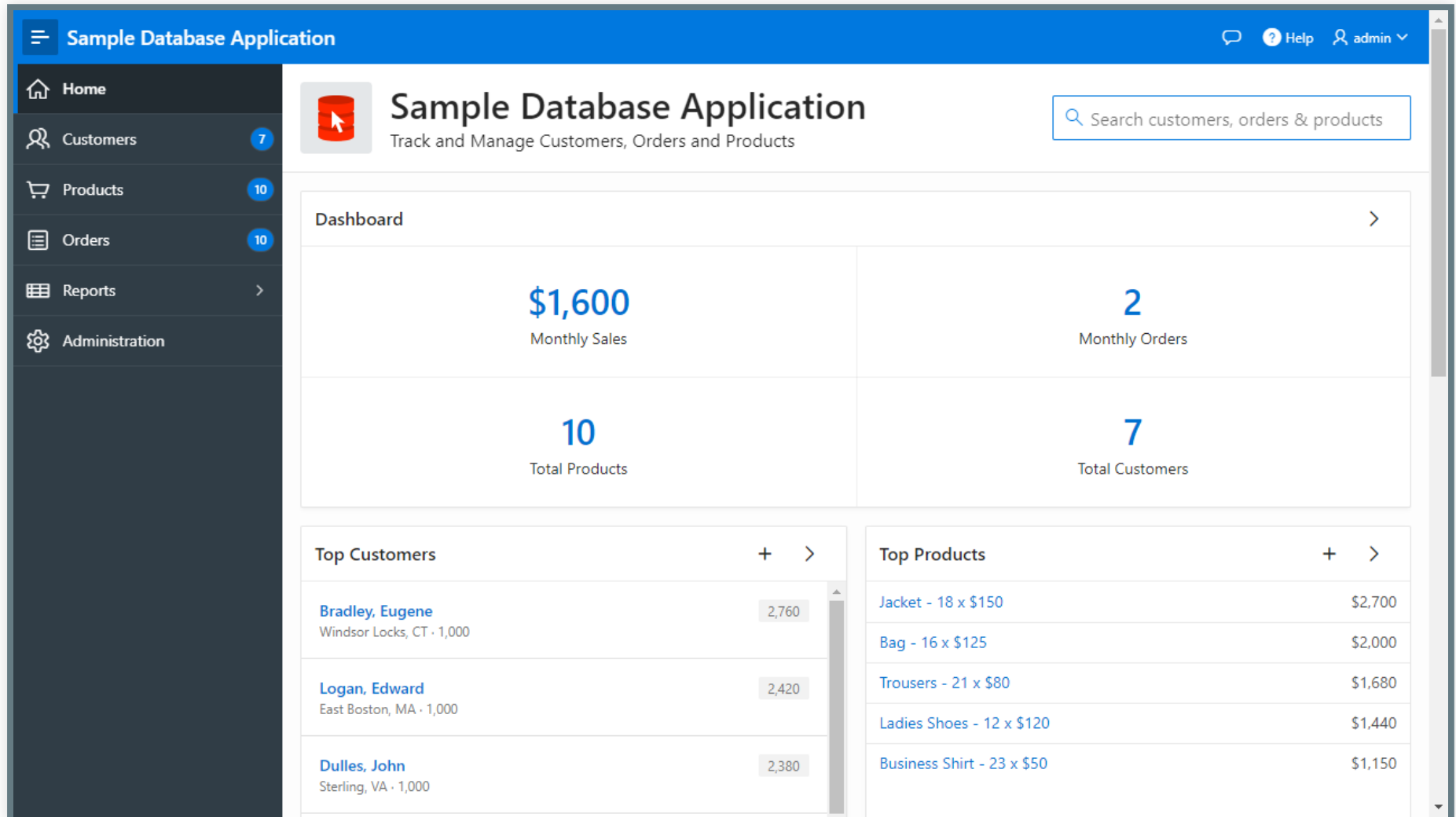
- Wenig konfigurierbar
- Enttäuscht für Aufbau Quellcode-Repos
- Viel Nacharbeit erforderlich

ANMERKUNGEN SQL DEVELOPER

- Ist am übersichtlichsten
- Viele Formate für Datenexport (auch CSV)
- Umfangreich konfigurierbar
- Blain Carter: [CI/CD for Database Developers – Export Database Objects into Version Control](#)
- Nacharbeit erforderlich

PLEX

AUSGANGSBASIS: SAMPLE DB APP



MÖGLICHER ERSTEXPORT

```
WITH
  FUNCTION backapp RETURN BLOB IS
  BEGIN
    RETURN plex.to_zip(plex.backapp(
      p_app_id           => 100,
      p_include_object_ddl => true,
      p_include_templates => true,
      p_include_runtime_log => true,
      p_include_data      => true,
      p_data_table_name_like => 'DEMO_PRODUCT_INFO,DEMO_STATES'
    ));
  END backapp;

SELECT backapp FROM dual;
```

[Blog Post](#)

DEMO

Create Repo & Export App

SPEICHERN ALS ZIP

The screenshot shows the Oracle SQL Developer interface with a PL/SQL function named `backapp` in the main editor. The function returns a BLOB and is designed to export data from a table into a ZIP file. A red circle and the number '1' highlight the edit icon in the query result grid.

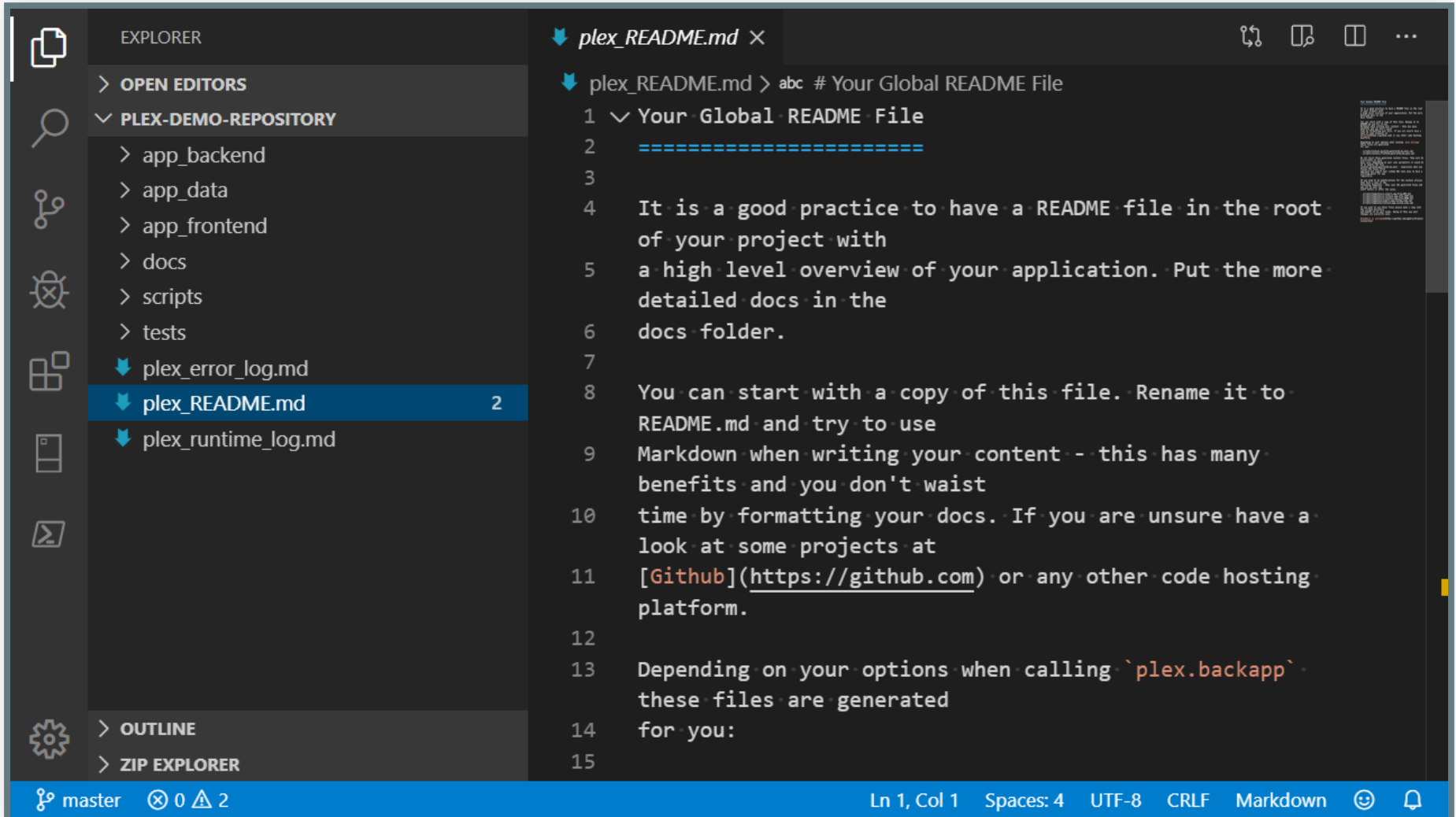
A 'View Value' dialog box is open, displaying the 'Saved Data' section. The 'Download' button is circled in red, and a red circle with the number '2' highlights the 'View As' options. The dialog shows the following data:

Saved Data	
Length	403893
Empty	false
Temporary	true
Open	false

The 'Local Data' section is also visible but empty.

The status bar at the bottom indicates 'Line 10 Column 20 | Insert | Windows: C'.

DAS ENTPACKTE ZIP FILE

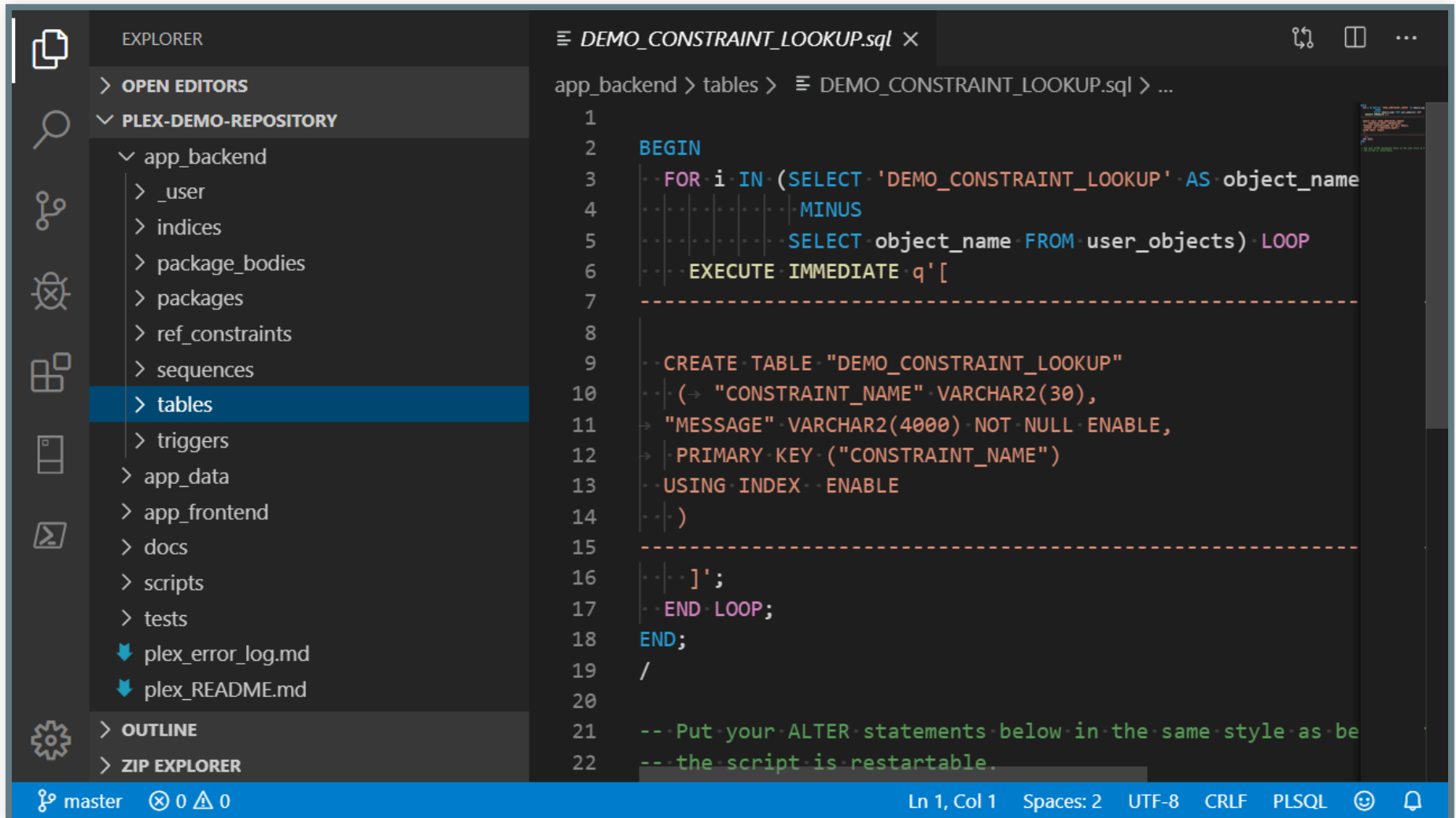


The screenshot shows the Visual Studio Code interface with a dark theme. On the left, the Explorer sidebar is open, showing a file tree for a repository named 'PLEX-DEMO-REPOSITORY'. The tree includes folders 'app_backend', 'app_data', 'app_frontend', 'docs', 'scripts', and 'tests', and files 'plex_error_log.md', 'plex_README.md' (selected), and 'plex_runtime_log.md'. Below the Explorer, the Outline and Zip Explorer sections are visible. The main editor area displays the 'plex_README.md' file, which contains a global README template. The status bar at the bottom indicates the current file is 'plex_README.md' on line 1, column 1, with 2 spaces, using UTF-8 encoding and CRLF line endings. The status bar also shows the current branch is 'master' and the file is not tracked.

```
plex_README.md ×
plex_README.md > abc # Your Global README File
1 < Your Global README File
2 =====
3
4 It is a good practice to have a README file in the root
5 of your project with
6 a high level overview of your application. Put the more
7 detailed docs in the
8 docs folder.
9
10 You can start with a copy of this file. Rename it to
11 README.md and try to use
12 Markdown when writing your content - this has many
13 benefits and you don't waist
14 time by formatting your docs. If you are unsure have a
15 look at some projects at
16 [Github](https://github.com) or any other code hosting
17 platform.
18
19 Depending on your options when calling `plex.backapp`
20 these files are generated
21 for you:
```

master 0 2 Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Markdown

BACKEND



The screenshot shows an IDE interface with a dark theme. On the left is the 'EXPLORER' sidebar, which contains a tree view of a project named 'PLEX-DEMO-REPOSITORY'. Under the 'app_backend' folder, the 'tables' folder is selected and highlighted in blue. The main editor area on the right displays a SQL script named 'DEMO_CONSTRAINT_LOOKUP.sql'. The script is a PL/SQL block that begins with a loop to iterate over objects in the 'user_objects' table, excluding the current script. It then creates a table named 'DEMO_CONSTRAINT_LOOKUP' with a primary key on 'CONSTRAINT_NAME'. The script ends with a comment indicating where ALTER statements should be placed.

```
1
2 BEGIN
3   FOR i IN (SELECT 'DEMO_CONSTRAINT_LOOKUP' AS object_name
4             MINUS
5             SELECT object_name FROM user_objects) LOOP
6     EXECUTE IMMEDIATE q'[
7
8
9     CREATE TABLE "DEMO_CONSTRAINT_LOOKUP"
10    (
11      "CONSTRAINT_NAME" VARCHAR2(30),
12      "MESSAGE" VARCHAR2(4000) NOT NULL ENABLE,
13      PRIMARY KEY ("CONSTRAINT_NAME")
14      USING INDEX ENABLE
15    )
16  ]';
17 END LOOP;
18 END;
19 /
20
21 -- Put your ALTER statements below in the same style as be
22 -- the script is restartable.
```

The status bar at the bottom indicates the current file is 'master', there are 0 errors and 0 warnings, and the cursor is at line 1, column 1. The encoding is UTF-8, line endings are CRLF, and the language is PLSQL.

KATALOGDATEN

The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with icons for Explorer, Search, Source Control, Run and Debug, Extensions, Remote Explorer, and Testing. The Explorer panel is active, showing a project structure under 'PLEX-DEMO-REPOSITORY'. The 'app_data' folder is expanded, showing 'DEMO_PRODUCT_INFO.csv' and 'DEMO_STATES.csv'. The 'DEMO_STATES.csv' file is selected and its content is displayed in the main editor area. The file content is a CSV with two columns: 'ST' and 'STATE_NAME', listing US states from Alaska to Maryland. The status bar at the bottom shows 'master', '0' errors/warnings, 'CSVLint' as the active linter, 'RBQL' as the language, and various editor settings like 'Align', 'Rainbow OFF', 'Ln 1, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'CSV', and a search icon.

EXPLORER

> OPEN EDITORS

▼ PLEX-DEMO-REPOSITORY

- > app_backend
- ▼ app_data
 - DEMO_PRODUCT_INFO.csv
 - DEMO_STATES.csv
- > app_frontend
- > docs
- > scripts
- > tests
- plex_error_log.md
- plex_README.md
- plex_runtime_log.md

> OUTLINE

> ZIP EXPLORER

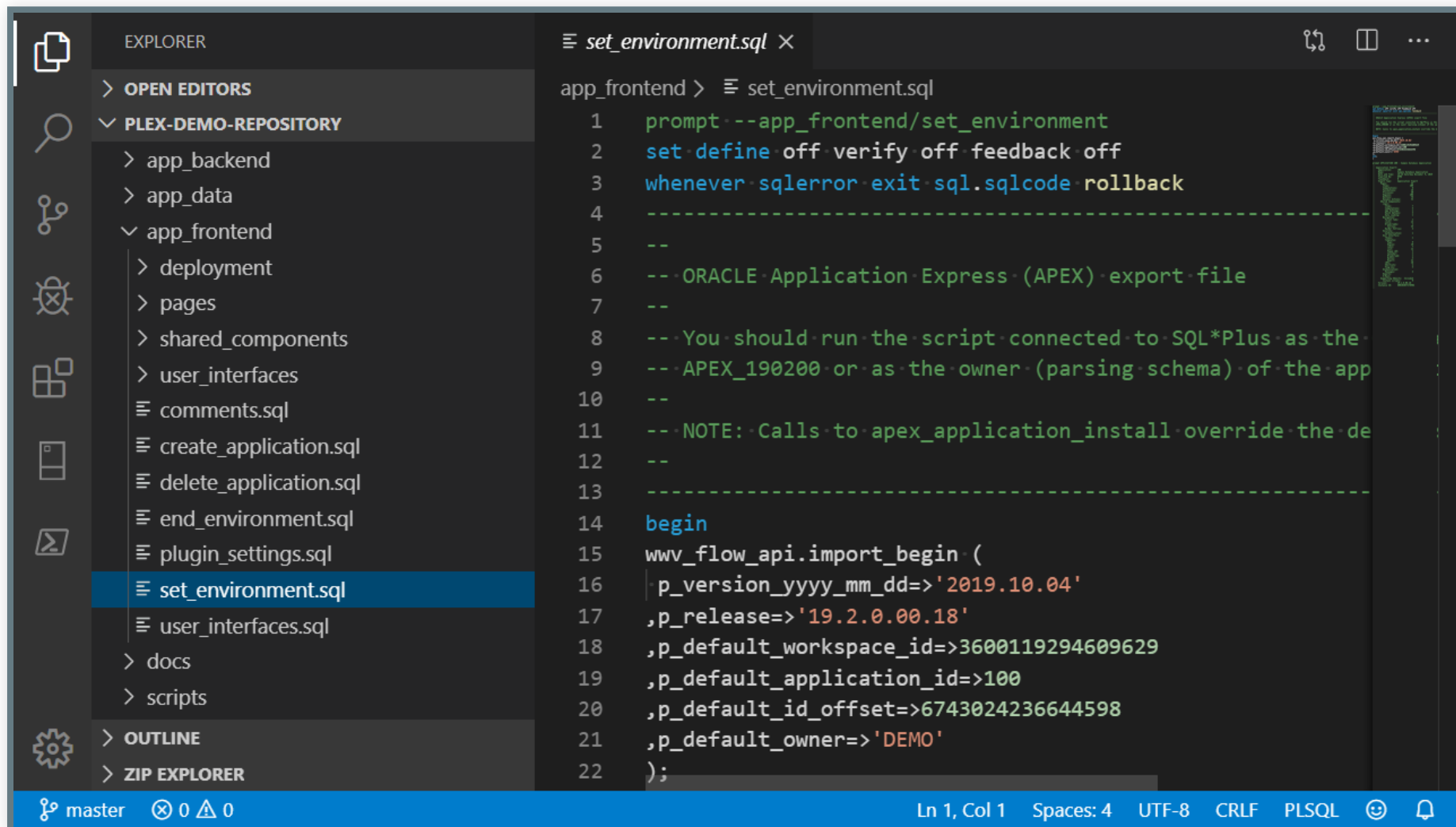
DEMO_STATES.csv

app_data > DEMO_STATES.csv

```
1 ST,STATE_NAME
2 AK,ALASKA
3 AL,ALABAMA
4 AR,ARKANSAS
5 AZ,ARIZONA
6 CA,CALIFORNIA
7 CO,COLORADO
8 CT,CONNECTICUT
9 DC,DISTRICT OF COLUMBIA
10 DE,DELAWARE
11 FL,FLORIDA
12 GA,GEORGIA
13 HI,HAWAII
14 IA,IOWA
15 ID,IDAHO
16 IL,ILLINOIS
17 IN,INDIANA
18 KS,KANSAS
19 KY,KENTUCKY
20 LA,LOUISIANA
21 MA,MASSACHUSETTS
22 MD,MARYLAND
```

master 0 0 CSVLint RBQL Align Rainbow OFF Ln 1, Col 1 Spaces: 4 UTF-8 CRLF CSV

FRONTEND ZERLEGT



The image shows a screenshot of an IDE (Integrated Development Environment) with a dark theme. On the left is the EXPLORER sidebar, and on the right is the main editor window.

EXPLORER Sidebar:

- EXPLORED
- OPEN EDITORS
- PLEX-DEMO-REPOSITORY
 - app_backend
 - app_data
 - app_frontend
 - deployment
 - pages
 - shared_components
 - user_interfaces
 - comments.sql
 - create_application.sql
 - delete_application.sql
 - end_environment.sql
 - plugin_settings.sql
 - set_environment.sql** (selected)
 - user_interfaces.sql
 - docs
 - scripts
- OUTLINE
- ZIP EXPLORER

Main Editor Window:

The editor shows a file named `set_environment.sql`. The content of the file is as follows:

```
1  prompt --app_frontend/set_environment
2  set define off verify off feedback off
3  whenever sqlerror exit sql.sqlcode rollback
4  -----
5  --
6  --ORACLE Application Express (APEX) export file
7  --
8  --You should run the script connected to SQL*Plus as the
9  --APEX_190200 or as the owner (parsing schema) of the app
10 --
11 --NOTE: Calls to apex_application_install override the de
12 --
13 -----
14 begin
15 wwv_flow_api.import_begin (
16   p_version_yyyy_mm_dd=>'2019.10.04'
17   ,p_release=>'19.2.0.00.18'
18   ,p_default_workspace_id=>3600119294609629
19   ,p_default_application_id=>100
20   ,p_default_id_offset=>6743024236644598
21   ,p_default_owner=>'DEMO'
22 );
```

The status bar at the bottom indicates: master, 0 errors, 0 warnings, Ln 1, Col 1, Spaces: 4, UTF-8, CRLF, PLSQL, and icons for search and help.

BACKEND MASTER SCRIPT

EXPLORER

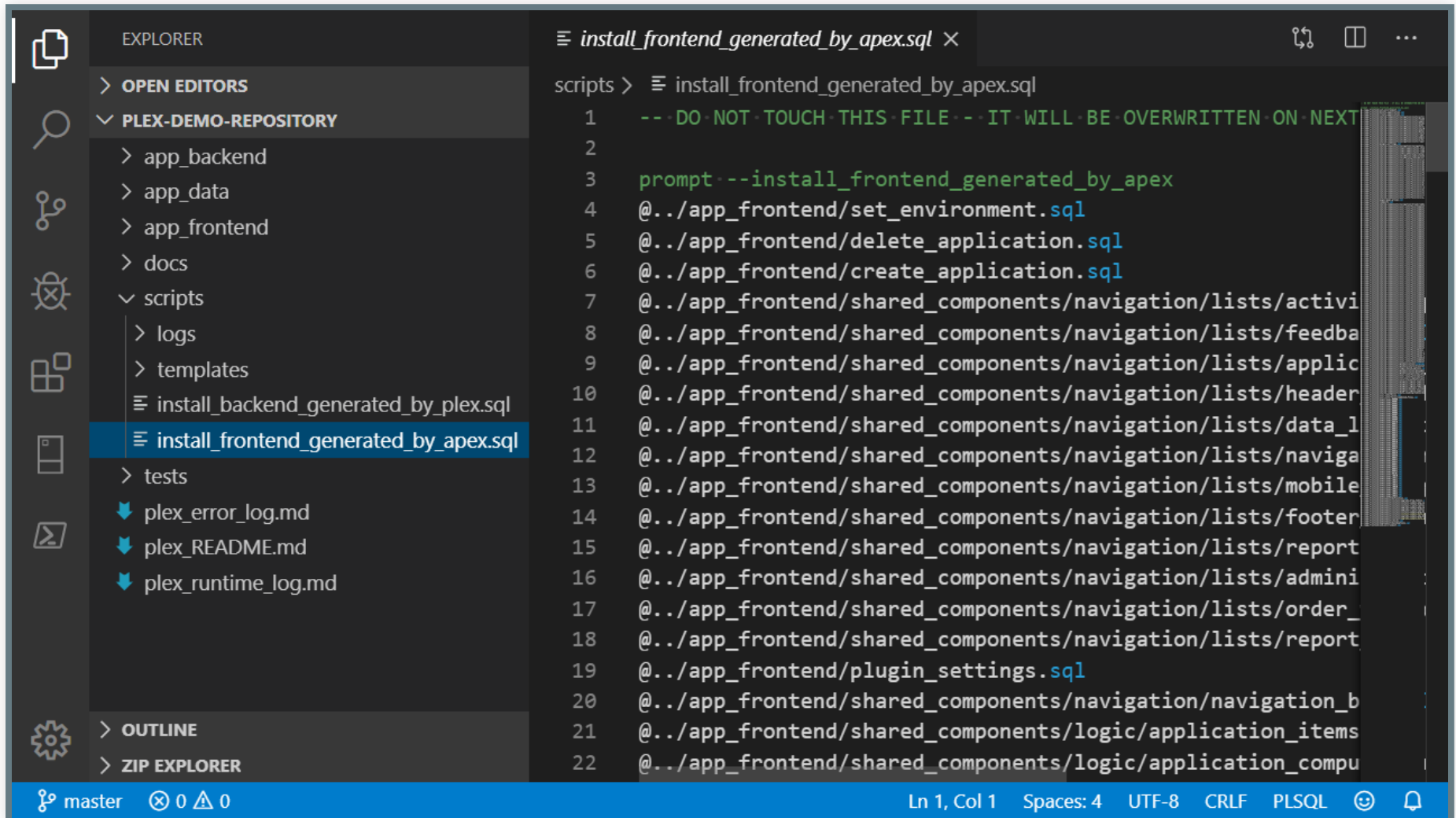
- > OPEN EDITORS
- ▼ PLEX-DEMO-REPOSITORY
 - > app_backend
 - > app_data
 - > app_frontend
 - > docs
 - ▼ scripts
 - > logs
 - > templates
 - ≡ install_backend_generated_by_plex.sql
 - ≡ install_frontend_generated_by_apex.sql
 - > tests
 - 📄 plex_error_log.md
 - 📄 plex_README.md
 - 📄 plex_runtime_log.md
- > OUTLINE
- > ZIP EXPLORER

scripts > ≡ install_backend_generated_by_plex.sql

```
1  /*-A-T-T-E-N-T-I-O-N
2  DO NOT TOUCH THIS FILE or set the PLEX.BackApp parameter p
3  to false -- otherwise your changes would be overwritten on
4  call. It is recommended to export your object ddl only one
5  repository creation and then start to use the "files first
6  */
7
8  set define off verify off feedback off
9  whenever sqlerror exit sql.sqlcode rollback
10
11  prompt --install_backend_generated_by_plex
12
13  prompt --app_backend/sequences/DEMO_CUST_SEQ
14  @../app_backend/sequences/DEMO_CUST_SEQ.sql
15
16  prompt --app_backend/sequences/DEMO_ORDER_ITEMS_SEQ
17  @../app_backend/sequences/DEMO_ORDER_ITEMS_SEQ.sql
18
19  prompt --app_backend/sequences/DEMO_ORD_SEQ
20  @../app_backend/sequences/DEMO_ORD_SEQ.sql
21
22  prompt --app_backend/sequences/DEMO_PROD_SEQ
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF PLSQL

FRONTEND MASTER SCRIPT



The image shows a screenshot of an IDE interface. On the left is the 'EXPLORER' panel, which displays a file tree for a project named 'PLEX-DEMO-REPOSITORY'. The tree includes folders like 'app_backend', 'app_data', 'app_frontend', 'docs', 'scripts', 'logs', 'templates', 'tests', and files like 'plex_error_log.md', 'plex_README.md', and 'plex_runtime_log.md'. The file 'install_frontend_generated_by_apex.sql' is selected and highlighted in blue. On the right is the SQL script editor, showing the content of the selected file. The script starts with a comment: '-- DO NOT TOUCH THIS FILE -- IT WILL BE OVERWRITTEN ON NEXT'. It then contains a series of SQL commands, including a prompt to install the frontend, and several '@../app_frontend/' commands that execute various SQL files like 'set_environment.sql', 'delete_application.sql', 'create_application.sql', and various navigation and application logic files. The status bar at the bottom indicates the current file is 'master', there are 0 errors and 0 warnings, and the editor is at line 1, column 1, with 4 spaces, UTF-8 encoding, CRLF line endings, and PLSQL dialect.

EXPLORER

> OPEN EDITORS

▼ PLEX-DEMO-REPOSITORY

- > app_backend
- > app_data
- > app_frontend
- > docs
- ▼ scripts
 - > logs
 - > templates
 - ≡ install_backend_generated_by_plex.sql
 - ≡ install_frontend_generated_by_apex.sql
 - > tests
- ▼ plex_error_log.md
- ▼ plex_README.md
- ▼ plex_runtime_log.md

> OUTLINE

> ZIP EXPLORER

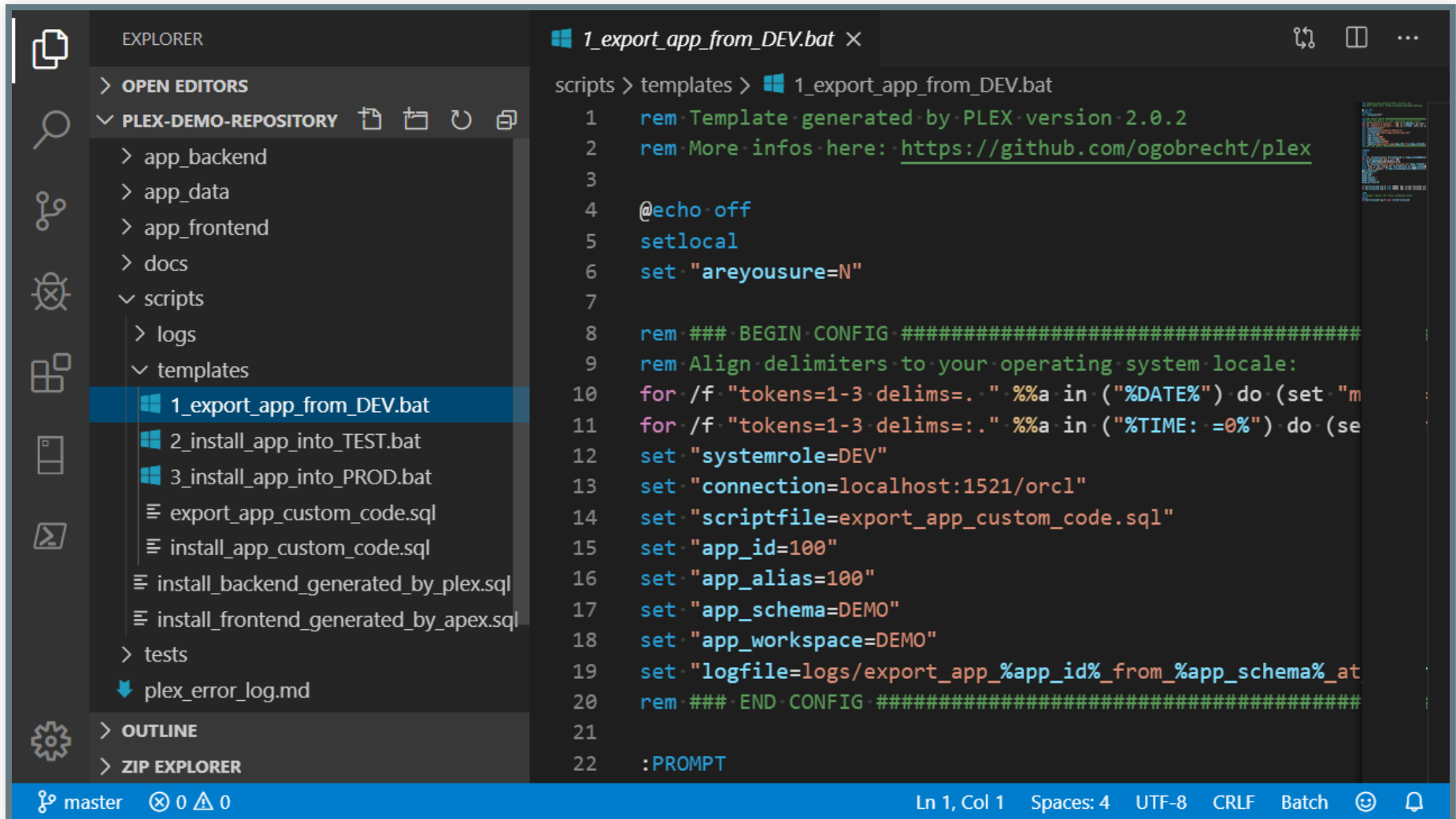
≡ install_frontend_generated_by_apex.sql ×

scripts > ≡ install_frontend_generated_by_apex.sql

```
1  -- DO NOT TOUCH THIS FILE -- IT WILL BE OVERWRITTEN ON NEXT
2
3  prompt --install_frontend_generated_by_apex
4  @../app_frontend/set_environment.sql
5  @../app_frontend/delete_application.sql
6  @../app_frontend/create_application.sql
7  @../app_frontend/shared_components/navigation/lists/activi
8  @../app_frontend/shared_components/navigation/lists/feedba
9  @../app_frontend/shared_components/navigation/lists/applic
10 @../app_frontend/shared_components/navigation/lists/header
11 @../app_frontend/shared_components/navigation/lists/data_l
12 @../app_frontend/shared_components/navigation/lists/naviga
13 @../app_frontend/shared_components/navigation/lists/mobile
14 @../app_frontend/shared_components/navigation/lists/footer
15 @../app_frontend/shared_components/navigation/lists/report
16 @../app_frontend/shared_components/navigation/lists/admini
17 @../app_frontend/shared_components/navigation/lists/order_
18 @../app_frontend/shared_components/navigation/lists/report
19 @../app_frontend/plugin_settings.sql
20 @../app_frontend/shared_components/navigation/navigation_b
21 @../app_frontend/shared_components/logic/application_items
22 @../app_frontend/shared_components/logic/application_compu
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF PLSQL

DEPLOYMENT TEMPLATES

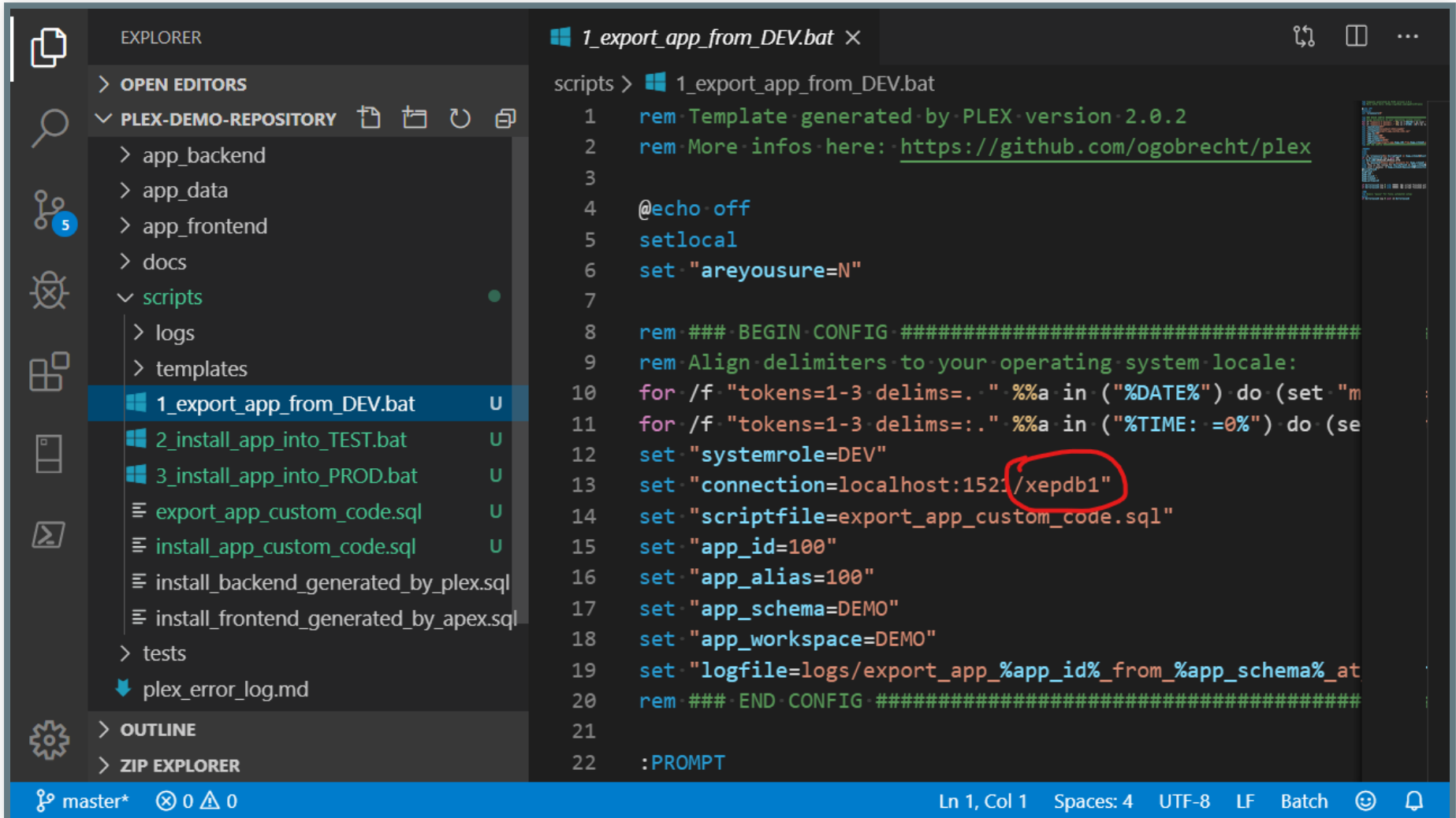


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Editor window on the right. The Explorer sidebar displays the file structure of the 'PLEX-DEMO-REPOSITORY', with the 'scripts' folder expanded and the 'templates' subfolder selected. The file '1_export_app_from_DEV.bat' is highlighted. The Editor window shows the content of this batch file, which is a deployment template for a Plex application. The script includes comments about the version and a link to the GitHub repository, followed by configuration settings for the application, such as the system role, connection string, script file, app ID, app alias, app schema, and app workspace. The script also sets the log file path and ends with a prompt.

```
1  rem Template generated by PLEX version 2.0.2
2  rem More infos here: https://github.com/ogobrecht/plex
3
4  @echo off
5  setlocal
6  set "areyousure=N"
7
8  rem ### BEGIN CONFIG #####
9  rem Align delimiters to your operating system locale:
10 for /f "tokens=1-3 delims=." %%a in ("%DATE%") do (set "m
11 for /f "tokens=1-3 delims=." %%a in ("%TIME: =0%") do (se
12 set "systemrole=DEV"
13 set "connection=localhost:1521/orcl"
14 set "scriptfile=export_app_custom_code.sql"
15 set "app_id=100"
16 set "app_alias=100"
17 set "app_schema=DEMO"
18 set "app_workspace=DEMO"
19 set "logfile=logs/export_app_%app_id%_from_%app_schema%_at
20 rem ### END CONFIG #####
21
22 :PROMPT
```

master 0 0 Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Batch

TEMPLATES KOPIERT & ANGEPASST



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the file structure of a project named 'PLEX-DEMO-REPOSITORY'. The 'scripts' folder is expanded, showing several files, with '1_export_app_from_DEV.bat' selected. The main editor area displays the content of this batch file. The script is a Windows batch file that sets various environment variables and configurations for a Plex application. A red circle highlights the 'connection' variable, which is set to 'localhost:1521/xepdb1'.

EXPLORER

> OPEN EDITORS

✓ PLEX-DEMO-REPOSITORY

- > app_backend
- > app_data
- > app_frontend
- > docs
- ✓ scripts
 - > logs
 - > templates
 - 1_export_app_from_DEV.bat U
 - 2_install_app_into_TEST.bat U
 - 3_install_app_into_PROD.bat U
 - ≡ export_app_custom_code.sql U
 - ≡ install_app_custom_code.sql U
 - ≡ install_backend_generated_by_plex.sql
 - ≡ install_frontend_generated_by_apex.sql
- > tests
- plex_error_log.md

> OUTLINE

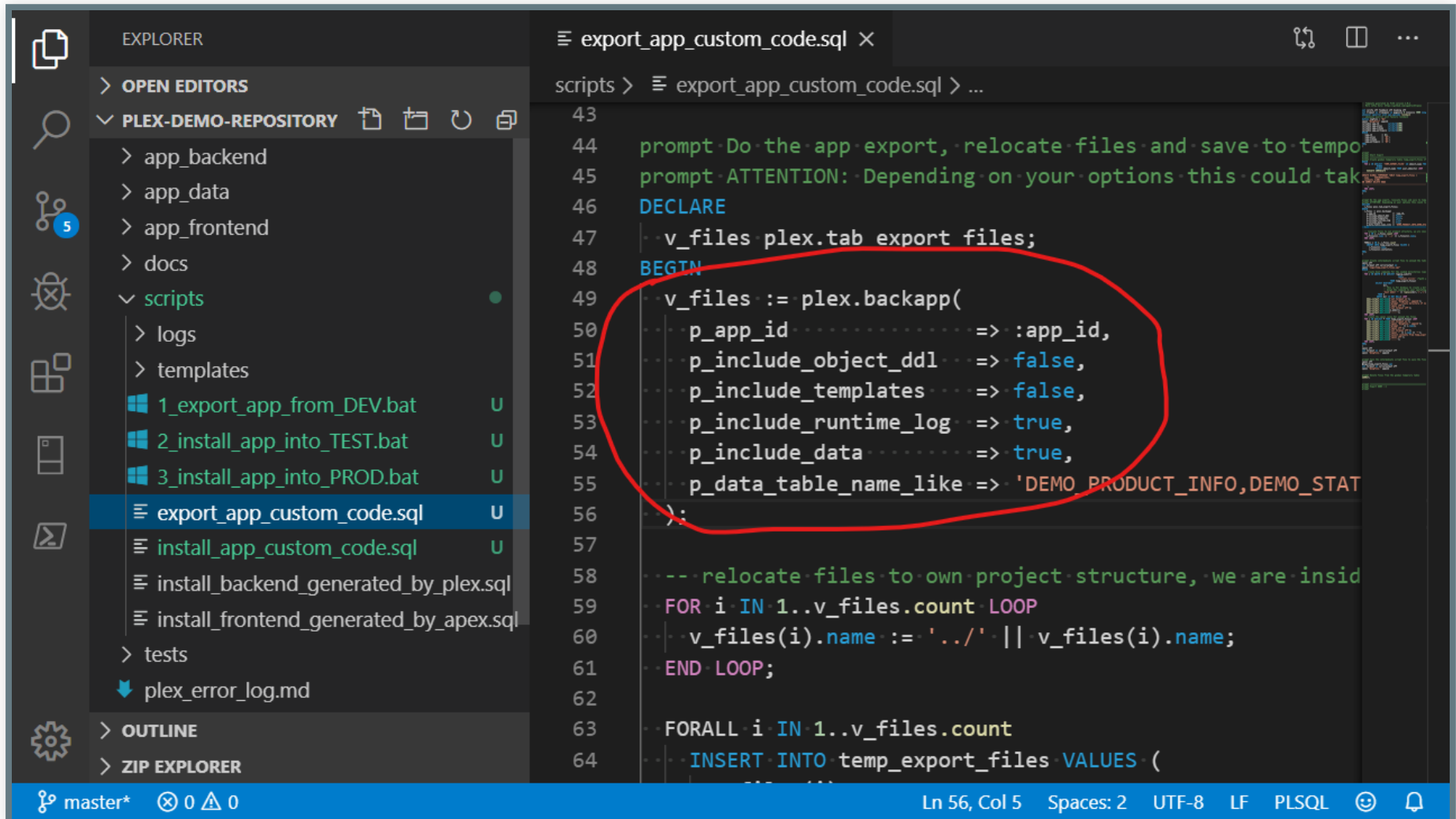
> ZIP EXPLORER

1_export_app_from_DEV.bat

```
scripts > 1_export_app_from_DEV.bat
1  rem Template generated by PLEX version 2.0.2
2  rem More infos here: https://github.com/ogobrecht/plex
3
4  @echo off
5  setlocal
6  set "areyousure=N"
7
8  rem ### BEGIN CONFIG #####
9  rem Align delimiters to your operating system locale:
10 for /f "tokens=1-3 delims=." %%a in ("%DATE%") do (set "m
11 for /f "tokens=1-3 delims=." %%a in ("%TIME: =0%") do (se
12 set "systemrole=DEV"
13 set "connection=localhost:1521/xepdb1"
14 set "scriptfile=export_app_custom_code.sql"
15 set "app_id=100"
16 set "app_alias=100"
17 set "app_schema=DEMO"
18 set "app_workspace=DEMO"
19 set "logfile=logs/export_app_%app_id%_from_%app_schema%.at
20 rem ### END CONFIG #####
21
22 :PROMPT
```

master* 0 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF Batch

ZUKÜNFTIGE EXPORTE KONFIGURIEREN



The screenshot displays an IDE interface with a file explorer on the left and a code editor on the right.

File Explorer (Left):

- EXPLORER
 - OPEN EDITORS
 - PLEX-DEMO-REPOSITORY
 - app_backend
 - app_data
 - app_frontend
 - docs
 - scripts
 - logs
 - templates
 - 1_export_app_from_DEV.bat
 - 2_install_app_into_TEST.bat
 - 3_install_app_into_PROD.bat
 - export_app_custom_code.sql** (selected)
 - install_app_custom_code.sql
 - install_backend_generated_by_plex.sql
 - install_frontend_generated_by_apex.sql
 - tests
 - plex_error_log.md
 - OUTLINE
 - ZIP EXPLORER

Code Editor (Right):

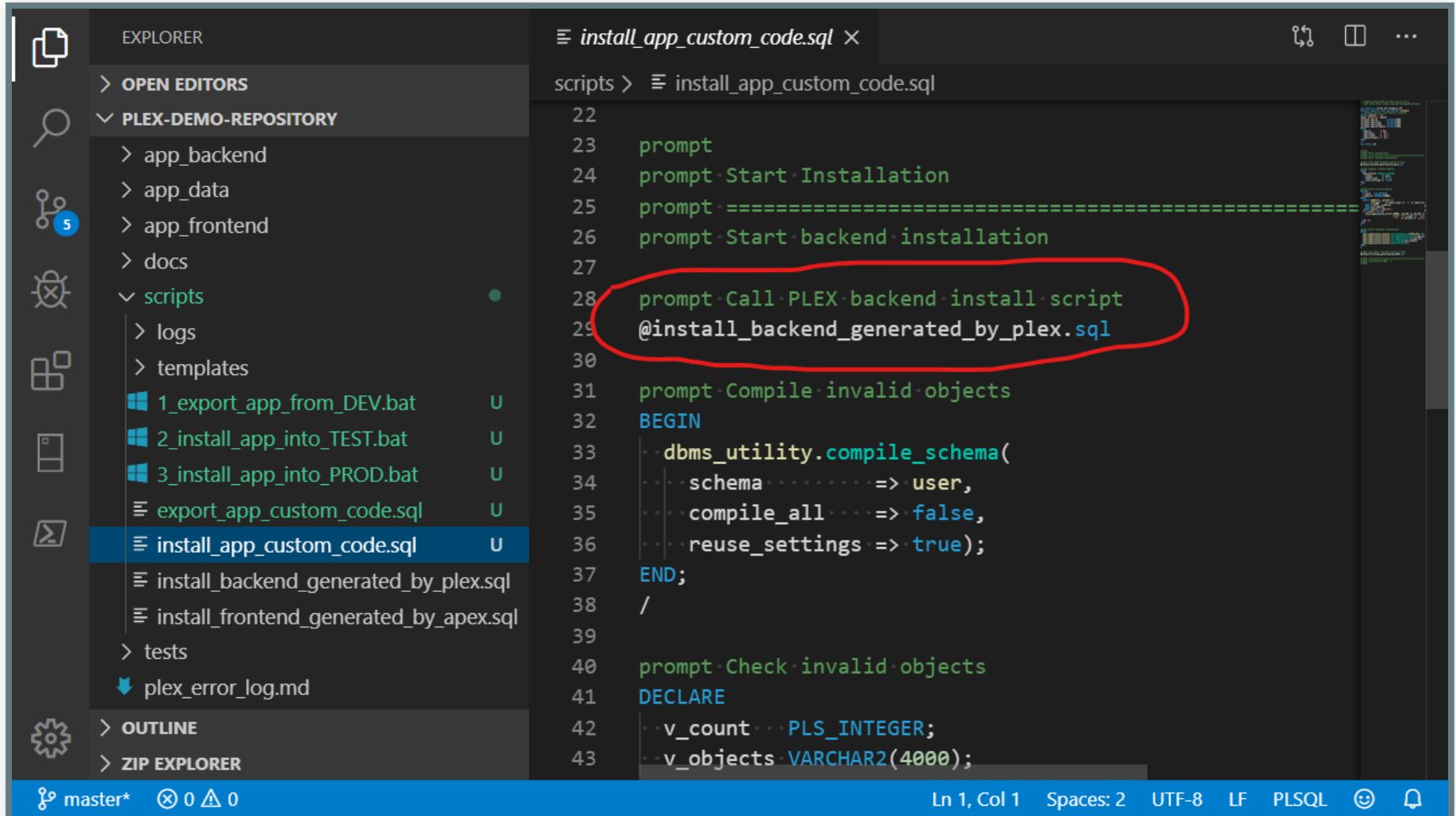
The code editor shows the file `export_app_custom_code.sql`. The code is as follows:

```
43
44 prompt Do the app export, relocate files and save to tempo
45 prompt ATTENTION: Depending on your options this could tak
46 DECLARE
47   v_files plex.tab export files;
48 BEGIN
49   v_files := plex.backapp(
50     p_app_id => :app_id,
51     p_include_object_ddl => false,
52     p_include_templates => false,
53     p_include_runtime_log => true,
54     p_include_data => true,
55     p_data_table_name_like => 'DEMO_PRODUCT_INFO,DEMO_STAT
56   );
57
58   ---relocate files to own project structure, we are insid
59   FOR i IN 1..v_files.count LOOP
60     v_files(i).name := '../' || v_files(i).name;
61   END LOOP;
62
63   FORALL i IN 1..v_files.count
64     INSERT INTO temp_export_files VALUES (
```

A red circle highlights the `plex.backapp()` function call and its parameters, indicating the configuration for future exports.

The status bar at the bottom shows: master* 0 0 Ln 56, Col 5 Spaces: 2 UTF-8 LF PLSQL

DEPLOYMENT MASTER SCRIPT 1/2

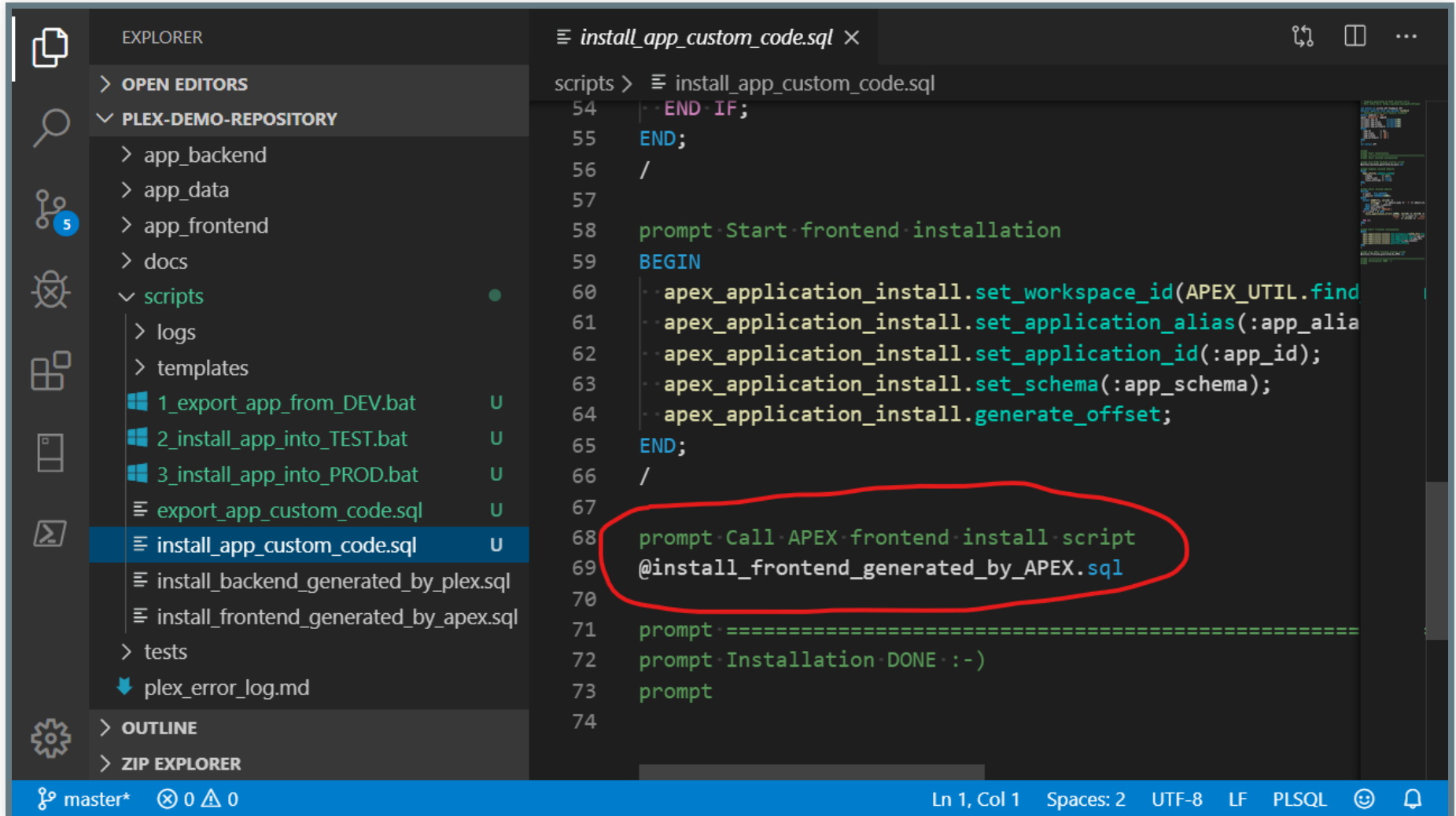


The screenshot displays an IDE interface with a dark theme. On the left, the 'EXPLORER' panel shows a project named 'PLEX-DEMO-REPOSITORY'. Under the 'scripts' folder, several files are listed, including 'install_app_custom_code.sql' which is currently selected. The main editor window shows the content of this file, which is a SQL script. A red circle highlights the line: `prompt·Call·PLEX·backend·install·script` followed by `@install_backend_generated_by_plex.sql`. The script includes prompts for installation, a call to compile invalid objects, and a declaration of variables for counting and object names.

```
22
23  prompt
24  prompt·Start·Installation
25  prompt·=====
26  prompt·Start·backend·installation
27
28  prompt·Call·PLEX·backend·install·script
29  @install_backend_generated_by_plex.sql
30
31  prompt·Compile·invalid·objects
32  BEGIN
33      ·dbms_utility.compile_schema(
34      ····schema····=>·user,
35      ····compile_all····=>·false,
36      ····reuse_settings =>·true);
37  END;
38  /
39
40  prompt·Check·invalid·objects
41  DECLARE
42      ·v_count···PLS_INTEGER;
43      ·v_objects·VARCHAR2(4000);
```

The status bar at the bottom indicates the current file is 'master*', there are 0 errors and 0 warnings, and the cursor is at line 1, column 1. The encoding is UTF-8, line feed (LF), and the language is PLSQL.

DEPLOYMENT MASTER SCRIPT 2/2



EXPLORER

OPEN EDITORS

PLEX-DEMO-REPOSITORY

- app_backend
- app_data
- app_frontend
- docs
- scripts
 - logs
 - templates
 - 1_export_app_from_DEV.bat U
 - 2_install_app_into_TEST.bat U
 - 3_install_app_into_PROD.bat U
 - export_app_custom_code.sql U
 - install_app_custom_code.sql U**
 - install_backend_generated_by_plex.sql
 - install_frontend_generated_by_apex.sql
- tests
- plex_error_log.md

OUTLINE

ZIP EXPLORER

install_app_custom_code.sql

```
54  END IF;  
55  END;  
56  /  
57  
58  prompt Start frontend installation  
59  BEGIN  
60  apex_application_install.set_workspace_id(APEX_UTIL.find  
61  apex_application_install.set_application_alias(:app_alia  
62  apex_application_install.set_application_id(:app_id);  
63  apex_application_install.set_schema(:app_schema);  
64  apex_application_install.generate_offset;  
65  END;  
66  /  
67  
68  prompt Call APEX frontend install script  
69  @install_frontend_generated_by_APEX.sql  
70  
71  prompt =====  
72  prompt Installation DONE :-)  
73  prompt  
74
```

master* 0 0 Ln 1, Col 1 Spaces: 2 UTF-8 LF PLSQL

DDL

AB HIER DATEIBASIERTES ARBEITEN

- Keine Änderungen über Klickibunti-Tools
- Alle Änderungen per Skript
- Nur noch deklarativen Code exportieren
 - APEX-Frontend
 - ORDS-REST-Services
 - ...

WIEDERANLAUFFÄHIGKEIT

The screenshot shows an IDE interface with a file explorer on the left and a code editor on the right. The file explorer, titled 'EXPLORER', shows a project named 'PLEX-DEMO-REPOSITORY' with a folder 'app_backend' containing several SQL files. The file 'DEMO_STATES.sql' is selected and highlighted in blue. The code editor, titled 'DEMO_STATES.sql', shows the following SQL script:

```
1
2 BEGIN
3   --FOR i IN (SELECT 'DEMO_STATES' AS object_name FROM dual
4   --           MINUS
5   --           SELECT object_name FROM user_objects) LOOP
6   --   EXECUTE IMMEDIATE q'[
7   -----
8
9   --CREATE TABLE "DEMO_STATES"
10  --  ( "ST" VARCHAR2(30),
11  --    "STATE_NAME" VARCHAR2(30)
12  --  )
13  -----
14  -- ];
15  --END LOOP;
16 END;
17 /
18
19 --Put your ALTER statements below in the same style as be
20 --the script is restartable.
21
22
```

The status bar at the bottom indicates the current file is 'master*', there are 0 errors and 0 warnings, and the cursor is at line 1, column 1. The status bar also shows 'Spaces: 2', 'UTF-8', 'CRLF', 'PLSQL', and icons for a smiley face and a bell.

DIE IDEE HINTER DER SKRIPTEREI

- Agile DB-Entwicklung
- Jede Änderung ist eine Migration
 - [Wikipedia: Schema migration](#)
 - [Artikel Samuel Nitsche](#)
 - [Artikel Martin Fowler](#)

Unser Ansatz ist nur eine mögliche Ausprägung der Idee

GESCHWINDIGKEIT

MEHRARBEIT, DIE SICH AUSZAHLT

- Nur Skripte
- Kein manueller App Export/Import
- Alle Skripte wiederaanlauffähig
- Gesamtablauf getestet
- Reduzierte „Deployment Pain“

DEMO

APEX Export & Deployment

APROPOS DEPLOYMENT PAIN

- Jede Umgebung ist individuell
- Mit den Skripten anfangen
- Kleine Schritte
- Immer besser werden
- Nicht stehenbleiben

MEHR TOOLS

GIT VERSUS SVN

- Git ist schneller
- SVN braucht weniger Platz
- Git funktioniert offline
- SVN Rechteverwaltung ist flexibler
- Entscheidungshilfe: [Artikel zum Thema](#)
- Tipp Windows Server: [Git](#), [SVN](#)

GITHUB DESKTOP

- Multi-Plattform (Linux in Arbeit)
- Reduziert auf das Wesentliche
- Übersichtlich
- Funktioniert mit eigenen Git-Servern
- [Homepage](#)

GITHUB DESKTOP

The screenshot displays the GitHub Desktop application window. The top menu bar includes File, Edit, View, Repository, Branch, and Help. Below the menu, the 'Current repository' is set to 'plex' and the 'Current branch' is 'development'. A 'Fetch origin' button indicates the last fetch was 'just now'.

The main interface is divided into three sections:

- Left Panel (Commit History):** Lists recent commits with their titles, authors, and dates. The selected commit is 'new build script' by Ottmar Gobrecht, committed on Sep 2, 2019.
- Center Panel (File Changes):** Shows a list of 10 changed files. The selected file is 'README.md'.
- Right Panel (Diff View):** Displays the diff for the selected file, showing changes between the current branch and the selected commit.

Commit History:

Commit Title	Author	Date
new build script	Ottmar Gobrecht	committed Sep 2, 2019
fix scripts for ORDS modules	Ottmar Gobrecht	committed Aug 19, 2019
include ORDS modules in templates	Ottmar Gobrecht	committed Aug 19, 2019
include ORDS modules	ogobrecht	committed Aug 18, 2019
fix error on large APEX UI install files	ogobrecht	committed Aug 16, 2019
fixes #2	ogobrecht	committed Jul 9, 2019
Merge pull request #1 from ogobrecht/v...	Ottmar Gobrecht	committed Jun 20, 2019
correct doc examples	ogobrecht	committed Jun 20, 2019
update docs	ogobrecht	committed Jun 20, 2019

File Changes:

File	Changes
README.md	@@ -1,4 +1,4 @@
package.json	-<!-- DO NOT EDIT THIS FILE DIRECTLY - it is generated from source file PLEX.pks -->
plex_install.sql	+<!-- DO NOT EDIT THIS FILE DIRECTLY - it is generated from source file src/PLEX.pks -->
PLEX.pkb → src\PLEX.pkb	PL/SQL Export Utilities
PLEX.pks → src\PLEX.pks	=====
src\build.js	
lib\as_zip.sql → src\lib\as_zip.sql	
lib\as_zip.txt → src\lib\as_zip.txt	
src\plex_install.sql	
src\plex_uninstall.sql	

Diff View (README.md):

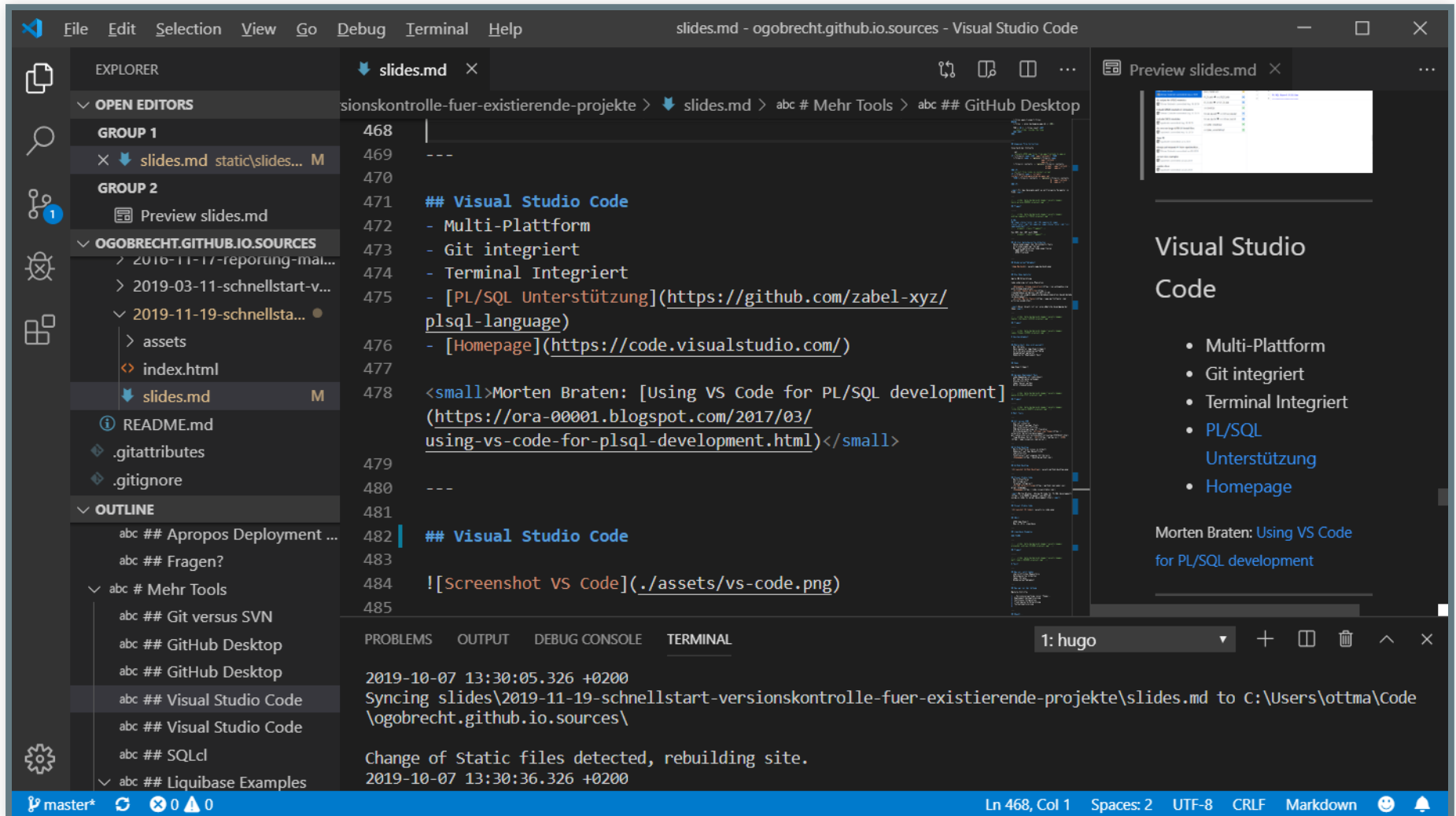
```
@@ -1,4 +1,4 @@
-<!-- DO NOT EDIT THIS FILE DIRECTLY - it is generated from source file PLEX.pks -->
+<!-- DO NOT EDIT THIS FILE DIRECTLY - it is generated from source file src/PLEX.pks -->
```

VISUAL STUDIO CODE

- Multi-Plattform
- Git integriert
- Terminal Integriert
- [PL/SQL Unterstützung](#)
- [Homepage](#)

Morten Braten: [Using VS Code for PL/SQL development](#)

VISUAL STUDIO CODE



LIQUIBASE

„Database schema change management“

- In SQLcl integriert (v19.2+)
- Jeff Smith: [Liquibase and SQLcl](#)
- www.liquibase.org

FAZIT

DIE REISE HAT BEGONNEN

1. ~~Versionsverwaltung~~
2. Deployment-Automatisierung
3. Continuous Integration
4. Trunk-basierte Entwicklung
5. Testautomatisierung
6. ...

Siehe Anhang „DevOps“

WEITERGEDACHT

Versionskontrolle nach DevOps meint
„Alle Produktionsartefakte“

- ~~Anwendungscode~~
- Build-Skripte
- Deployment Pipelines
- Systemkonfigurationen
- ...
- Kurz: Infrastruktur als Code

Übrigens: Diese Folien sind auch Code - geschrieben in Markdown

LESESTOFF 1

- Ottmar Gobrecht: **PLEX - PL/SQL Export Utilities & Schnellstart - Versionskontrolle für existierende Oracle-Projekte**
- Samuel Nitsche: **There is no clean (database) development without Version Control & “One does not simply update a database” – migration based database development**

LESESTOFF 2

- Blain Carter: [Tips to help PL/SQL developers get started with CI/CD & CI/CD for Database Developers – Export Database Objects into Version Control](#)
- Denis Savenko: [Oracle APEX and ORDS deployments automation](#)

LESESTOFF 3

- Martin Fowler: [Evolutionary Database Design](#)
- Jeff Smith: [Liquibase and SQLcl](#)
- Antti Kirmanen: [Git vs. Subversion \(SVN\): Welches Versionskontrollsystem sollten Sie nutzen?](#)
- Morten Braten: [Using VS Code for PL/SQL development](#)

THE END

ANHANG - DEVOPS

DEVOPS - STATISTIKEN

Leistungsstarke gegenüber leistungsschwachen Unternehmen laut [State of DevOps Report 2017](#)

- 46-mal häufigere Code Deployments
- 440-mal schneller von Commit zu Deployment
- 96-mal schnellere Wiederherstellung nach Ausfällen
- 5-mal niedrigere Ausfallrate bei Änderungen

KANN MAN DAS GLAUBEN?

Buchtip: [Das Mindset von DevOps - Accelerate](#)

- Performance der Softwarebereitstellung
- 24 Schlüsselkompetenzen in 5 Kategorien
- Wissenschaftlich belegt

Der Kern: Auf Kompetenzen, nicht auf Reife fokussieren.
Verbesserungen kontinuierlich vorantreiben.

DIE KOMPETENZ-KATEGORIEN

1. Continuous Delivery
2. Architektur
3. Produkt und Prozess
4. Lean Management und Monitoring
5. Kultur

CONTINUOUS DELIVERY-KOMPETENZEN

Die ersten Schritte

1. Versionsverwaltung
2. Deployment-Automatisierung
3. Continuous Integration
4. Trunk-basierte Entwicklung
5. Testautomatisierung
6. ...

KULTURELLE KOMPETENZEN - BEISPIEL ;-)

Kapitel 11: Führungskräfte und Manager, Transformationale Führung

*„Ermutigen Sie Ihre Belegschaft,
mindestens einmal im Jahr technische
Konferenzen zu besuchen und das dort
Gelernte für das gesamte Team
zusammenzufassen.“*

LESESTOFF

- Buchtip Heise Developer: [Das Mindset von DevOps - Accelerate - 24 Schlüsselkompetenzen, um leistungsstarke Technologieunternehmen zu entwickeln und zu skalieren](#)
- Gareth Rushgrove: [Macht DevOps Unternehmen erfolgreicher?](#)
- Puppet: State of DevOps Report [2017](#), [2018](#), [2019](#)