Week 8 Homework 4

Matt Rieser

SDEV 350 – Section 7980

December 16, 2018

Professor Ivan De Los Santos

Design and Justification of Design

HW4 Oracle 12c Database was created in AWS and SQL Developer was used to connect/design the database.

This is where I began my design. Inside the admin account I created tablespaces, a user profile for database users, and database users.

Admin user – SDEV350mrieser (for AWS database)

Tablespace creation – *table_audit*

Script:

DROP TABLESPACE table_audit INCLUDING CONTENTS;

CREATE TABLESPACE table_audit

DATAFILE SIZE 50M;

In this script I set the *SIZE* of the tablespaces above to *50M* so that users account created will have enough space inside. I also drop previously created tablespaces as to not cause an error when running the script.

Creation of Tablespaces is needed first before user profiles or users.

Temporary Tablespace creation – *TEMP_usersTable*

Script:

--TEMPORARY TABLESPACE CREATED

DROP TABLESPACE TEMP_usersTable;

CREATE TEMPORARY TABLESPACE TEMP_usersTable

TEMPFILE SIZE 50M;

In this script I set the *SIZE* of the temporary tablespaces above to *50M* so that users account created within will have enough space inside. I also drop previously created tablespaces as to not cause an error when running the script. I am using temporary tablespace to help speed performance.

Creation of Temporary Tablespaces is needed first before user profiles or users.

User Profile – user_profile

--CREATE PROFILE

DROP PROFILE user_profile CASCADE;

CREATE PROFILE user_profile

LIMIT

PASSWORD_REUSE_MAX 3

PASSWORD_REUSE_TIME 30

SESSIONS_PER_USER 2

CPU_PER_SESSION UNLIMITED

CPU_PER_CALL 3000

CONNECT_TIME 45

LOGICAL_READS_PER_SESSION DEFAULT

LOGICAL_READS_PER_CALL 1000

PRIVATE_SGA 15K

COMPOSITE_LIMIT 500000

FAILED_LOGIN_ATTEMPTS 3

PASSWORD_LIFE_TIME 30

PASSWORD_LOCK_TIME 7

PASSWORD_GRACE_TIME 10

PASSWORD_VERIFY_FUNCTION ora12c_verify_function;

Above I am creating a user profile with least privileges to be used as a profile for database users. Creation of a user profile is a quick way to limit user privileges, as you can use a PROFILE for multiple users. Notice that *PASSWORD_VERIFY_FUNCTION* is set to *ora12c_verify_function* this is for password complexity. Also notice that *FAILED_LOGIN_ATTEMPTS* is set to *3* and *PASSWORD_LOCK_TIME* is set to *7* minutes.

Database users – DBA1 (DBA = Database Admin)

Script:

-- USER CREATION DBA1

DROP USER DBA1;

CREATE USER DBA1 IDENTIFIED BY "Pa$$w0rd"

--PASSWORD EXPIRE

DEFAULT TABLESPACE "TABLE_AUDIT"

TEMPORARY TABLESPACE "TEMP_USERSTABLE"

PROFILE user_profile;

The creation of my database users. DBA1 is using tablespaces designed for the users side. Notice that their password is also set to expire after every login. The reason why it is commented out is because they were causing issues when trying to login into each database user. Notice that the user_profile is also being used for DBA1 and tablespace – table_audit.

Roles – Granting permission for Database users to connect, giving them Database privileges, and session privileges.

Script:

```
-- ROLES
GRANT CONNECT, "DBA" TO DBA1;

GRANT CREATE SESSION TO DBA1;
```

Design of DBA1_audit

DBA1_audit holds tables (BidRates2017, BidRates2018, Proposals2017, and Proposals2018) and audit tables (BR2017_audit, BR2018_audit, PP2017_audit, and PP2018_audit), a user profile with limiting privileges, 1 user with privileges to Select, Update, Insert and Delete tables - BidRates2017, BidRates2018, Proposals2017 and Proposals2018.

Dropping all tables – as to not cause errors like table with that name has already been created

Script:

--DROPPING TABLES

DROP TABLE BidRates2017;

DROP TABLE BR2017_audit;

DROP TRIGGER BidRates2017_audit;

DROP TABLE BidRates2018;
DROP TABLE BR2018_audit;
DROP TRIGGER BidRates2018_audit;

DROP TABLE Proposals2017;
DROP TABLE PP2017_audit;
DROP TRIGGER Proposals2017_audit;

DROP TABLE Proposals2018;
DROP TABLE PP2018_audit;
DROP TRIGGER Proposals2018_audit;

Tables – BidRates2017, BidRates2018, Proposals2017, and Proposals2018

Script:

```
-- CREATION OF TABLES
Create TABLE BidRates2017 (
    br_id int primary key,
    bid_id VARCHAR(20)
);
Create TABLE BidRates2018 (
   br_id int primary key,
   bid_id VARCHAR(20)
);
Create TABLE Proposals2017 (
    p_id int primary key,
    proposal_id VARCHAR(20)
```

```
);
Create TABLE Proposals2018 (
    p_id int primary key,
    proposal_id VARCHAR(20)
);
```

I went by the guide lines on tables. The parameters of each table are the only things that I designed. I gave each table an primary id and id (BidRate tables get a bid_id and Proposal tables get a proposal_id) parameters to act as it were in a real-life scenario.

Audit Tables – BR2017_audit, BR2018_audit, PP2017_audit, and PP2018_audit

Script:

```
-- CREATION OF AUDIT TABLES
CREATE TABLE BR2017_audit(
    new_id varchar (30),
    old_id varchar (30),
    user_name varchar (30),
    entry_date varchar (30),
    operation varchar (30)
);

CREATE TABLE BR2018_audit(
    new_id varchar (30),
    old_id varchar (30),
    user_name varchar (30),
    entry_date varchar (30),
    operation varchar (30)
);
```

```
CREATE TABLE PP2017_audit(
   new_id varchar (30),
   old_id varchar (30),
   user_name varchar (30),
   entry_date varchar (30),
   operation varchar (30)
);

CREATE TABLE PP2018_audit(
   new_id varchar (30),
   old_id varchar (30),
   user_name varchar (30),
   entry_date varchar (30),
   operation varchar (30)
);
```

The creation of these tables are used to hold the audit logs that are created when a user makes changes (delete, insert, update) to the tables - BidRates2017, BidRates2018, Proposals2017, and Proposals2018. The parameters of each table are used to display the change that occurred, the user that made it, the date they did it and the operation that they used.

User Profile

Script:

```
--CREATE PROFILE
DROP PROFILE user_profile CASCADE;
CREATE PROFILE user_profile
LIMIT
```

PASSWORD_REUSE_MAX 3

PASSWORD_REUSE_TIME 30

SESSIONS_PER_USER 2

CPU_PER_SESSION UNLIMITED

CPU_PER_CALL 3000

CONNECT_TIME 45

LOGICAL_READS_PER_SESSION DEFAULT

LOGICAL_READS_PER_CALL 1000

PRIVATE_SGA 15K

COMPOSITE_LIMIT 500000

FAILED_LOGIN_ATTEMPTS 3

PASSWORD_LIFE_TIME 30

PASSWORD_LOCK_TIME 7

PASSWORD_GRACE_TIME 10

PASSWORD_VERIFY_FUNCTION ora12c_verify_function;

Same user profile settings as in the sdev350mrieser account.

Creation of 1 user with privileges to Select, Update, Insert and Delete to BidRates2017, BidRates2018, Proposals2017, and Proposals2018

Script:

```
--CREATION OF AUDIT TRIGGERS FOR BidRates2017 TABLE
CREATE OR REPLACE TRIGGER BidRates2017_audit
BEFORE INSERT OR DELETE OR UPDATE ON BidRates2017
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(30);
```

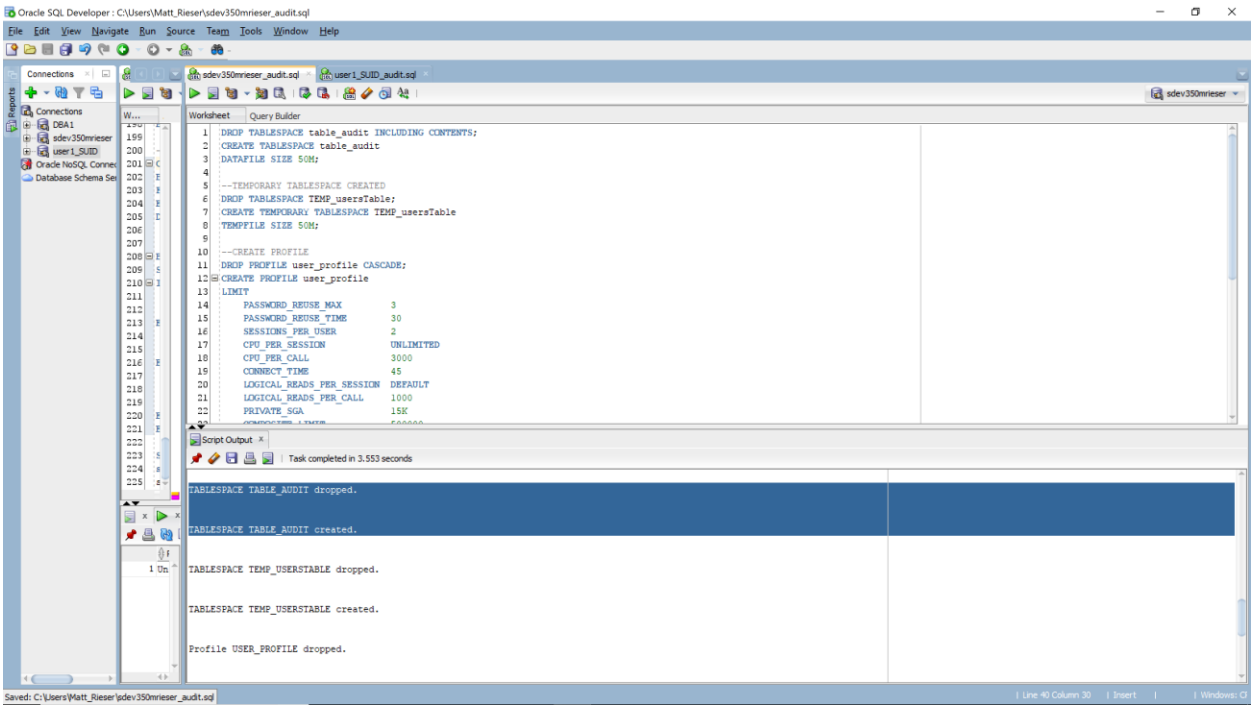
```
   v_date VARCHAR(30);
BEGIN
SELECT USER,to_char(sysdate, 'DD/MON/YYYY HH24:MI:SS') INTO v_user, v_date from
dual;
IF INSERTING THEN
   INSERT INTO BR2017_audit(new_id,old_id, user_name, entry_date, operation)
   VALUES (:NEW.bid_id, NULL, v_user, v_date, 'INSERT');
ELSIF DELETING THEN
   INSERT INTO BR2017_audit(new_id, old_id, user_name, entry_date, operation)
   VALUES (NULL, :OLD.bid_id, v_user, v_date, 'Delete');
ELSIF UPDATING THEN
   INSERT INTO BR2017_audit(new_id, old_id, user_name, entry_date, operation)
   VALUES (:NEW.bid_id, :OLD.bid_id, v_user, v_date, 'Update');

END IF;
END;

--CREATION OF AUDIT TRIGGERS FOR BidRates2018 TABLE
CREATE OR REPLACE TRIGGER BidRates2018_audit
BEFORE INSERT OR DELETE OR UPDATE ON BidRates2018
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(30);
   v_date VARCHAR(30);
BEGIN
SELECT USER,to_char(sysdate, 'DD/MON/YYYY HH24:MI:SS') INTO v_user, v_date from
dual;
IF INSERTING THEN
   INSERT INTO BR2018_audit(new_id,old_id, user_name, entry_date, operation)
```

```
   VALUES (:NEW.bid_id, NULL, v_user, v_date, 'INSERT');
ELSIF DELETING THEN
   INSERT INTO BR2018_audit(new_id, old_id, user_name, entry_date, operation)
   VALUES (NULL, :OLD.bid_id, v_user, v_date, 'Delete');
ELSIF UPDATING THEN
   INSERT INTO BR2018_audit(new_id, old_id, user_name, entry_date, operation)
   VALUES (:NEW.bid_id, :OLD.bid_id, v_user, v_date, 'Update');

END IF;
END;

--CREATION OF AUDIT TRIGGERS FOR Proposals2017 TABLE
CREATE OR REPLACE TRIGGER Proposals2017_audit
BEFORE INSERT OR DELETE OR UPDATE ON Proposals2017
FOR EACH ROW
ENABLE
DECLARE
   v_user VARCHAR(30);
   v_date VARCHAR(30);
BEGIN
SELECT USER,to_char(sysdate, 'DD/MON/YYYY HH24:MI:SS') INTO v_user, v_date from
dual;
IF INSERTING THEN
   INSERT INTO PP2017_audit(new_id,old_id, user_name, entry_date, operation)
   VALUES (:NEW.proposal_id, NULL, v_user, v_date, 'INSERT');
ELSIF DELETING THEN
   INSERT INTO PP2017_audit(new_id, old_id, user_name, entry_date, operation)
   VALUES (NULL, :OLD.proposal_id, v_user, v_date, 'Delete');
ELSIF UPDATING THEN
   INSERT INTO PP2017_audit(new_id, old_id, user_name, entry_date, operation)
```

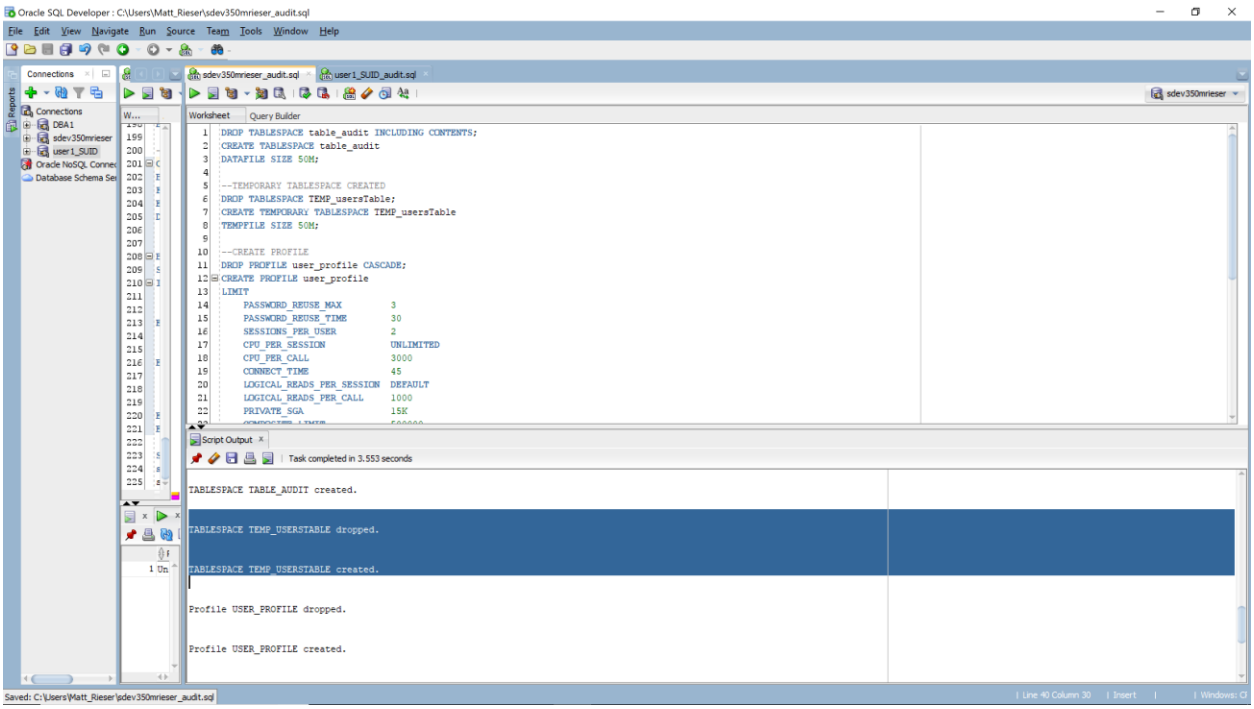
```
    VALUES (:NEW.proposal_id, :OLD.proposal_id, v_user, v_date, 'Update');


END IF;
END;

--CREATION OF AUDIT TRIGGERS FOR Proposals2018 TABLE
CREATE OR REPLACE TRIGGER Proposals2018_audit
BEFORE INSERT OR DELETE OR UPDATE ON Proposals2018
FOR EACH ROW
ENABLE
DECLARE
    v_user VARCHAR(30);
    v_date VARCHAR(30);
BEGIN
SELECT USER,to_char(sysdate, 'DD/MON/YYYY HH24:MI:SS') INTO v_user, v_date from dual;
IF INSERTING THEN
    INSERT INTO PP2018_audit(new_id,old_id, user_name, entry_date, operation)
    VALUES (:NEW.proposal_id, NULL, v_user, v_date, 'INSERT');
ELSIF DELETING THEN
    INSERT INTO PP2018_audit(new_id, old_id, user_name, entry_date, operation)
    VALUES (NULL, :OLD.proposal_id, v_user, v_date, 'Delete');
ELSIF UPDATING THEN
    INSERT INTO PP2018_audit(new_id, old_id, user_name, entry_date, operation)
    VALUES (:NEW.proposal_id, :OLD.proposal_id, v_user, v_date, 'Update');


END IF;
END;
```

I created 4 audit triggers (BidRates2017_audit, BidRates2018_audit, Proposals2017_audit, Proposals2018_audit) for tables – BidRates2017, BidRates2018, Proposals2017, and Proposals2018. Each trigger is used to record actions made by a user to bidrates and proposal tables. These actions include deleting, inserting, and updating data. To view any actions a user has made, all you have to do is select from either of the audit tables - BR2017_audit, BR2018_audit, PP2017_audit, and PP2018_audit.

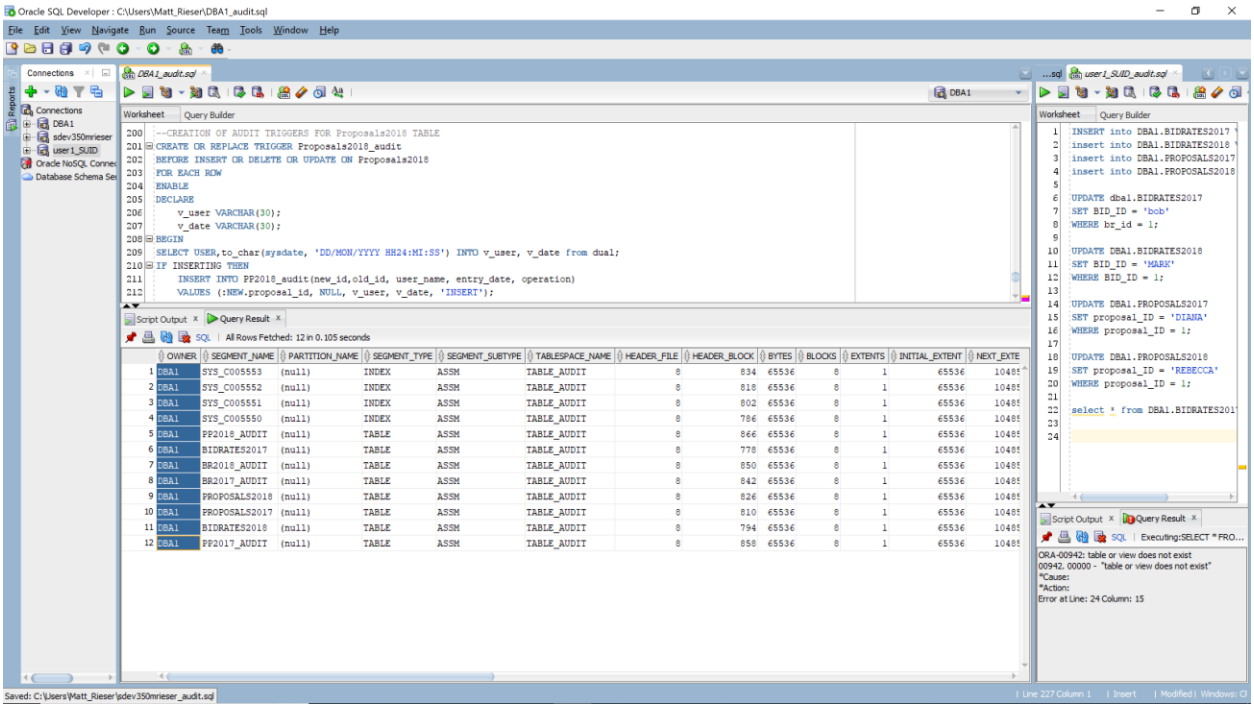
Creation and use of Tablespaces/Temporary Tablespaces



Above – Tablespace Table_audit successfully dropped and created

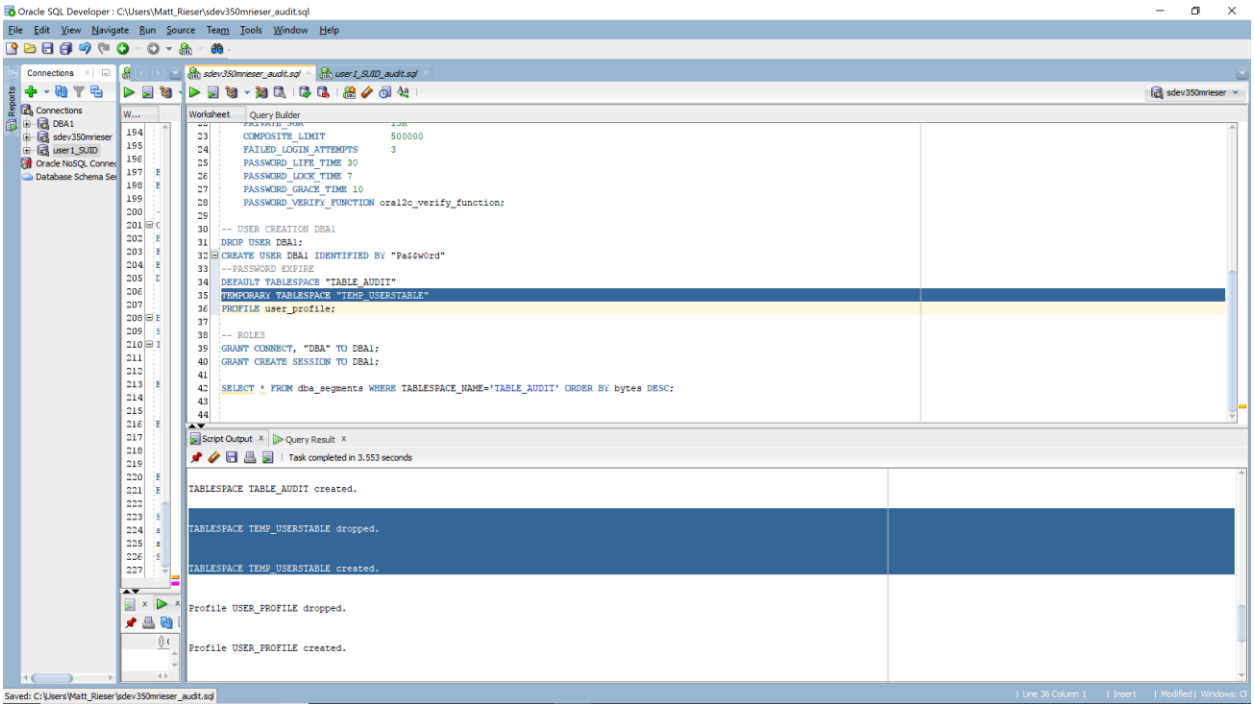


Above – Temporary Tablespaces – TEMP_usersTable successfully dropped and created



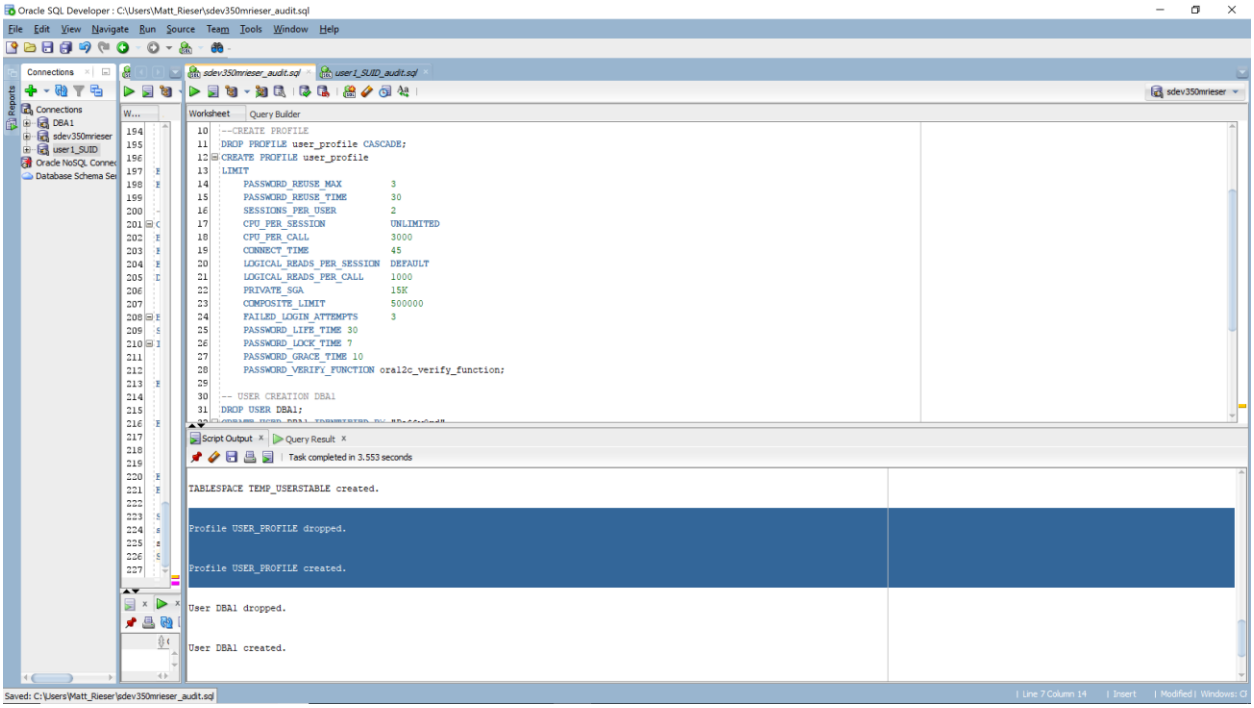
Above – uses the *SELECT * FROM dba_segments WHERE TABLESPACE_NAME='TABLE_AUDIT'*

ORDER BY bytes DESC; to display the use of TABLE_AUDIT

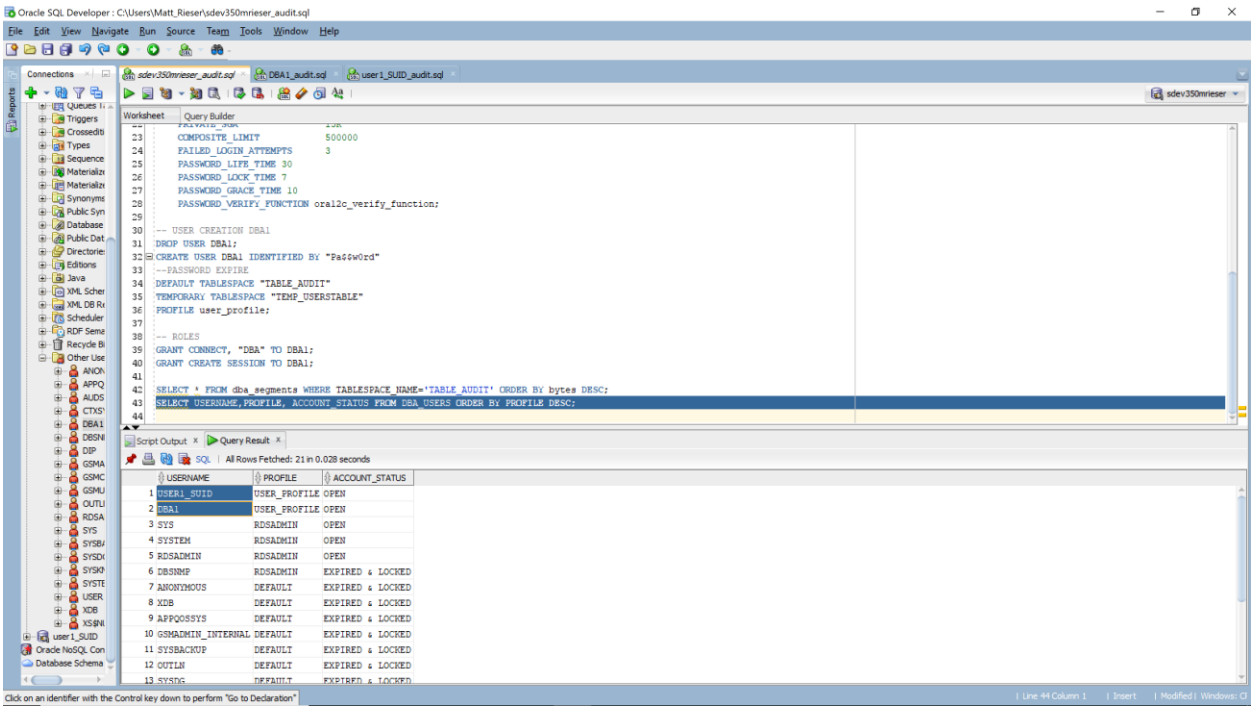


Above – Temporary Tablespace – TEMP_USERTABLE is being used by DBA1 and all its users

Creation and use of Profiles

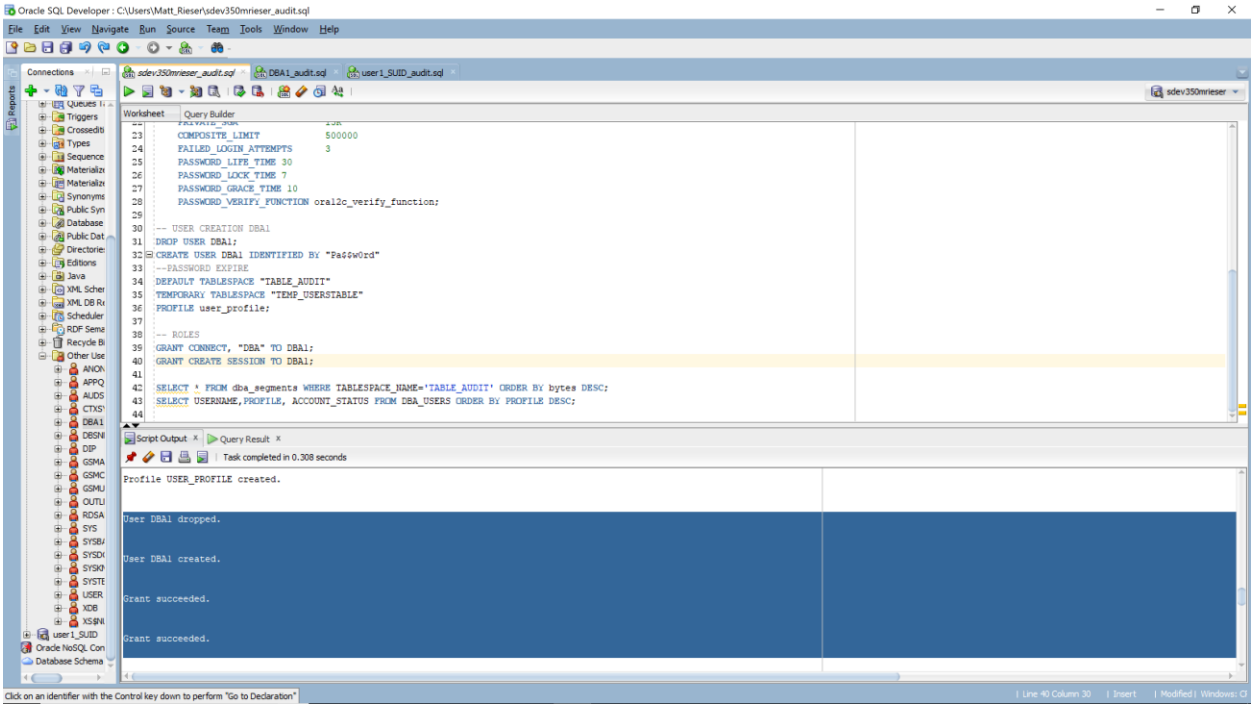


Above – shows the creation of user_profile

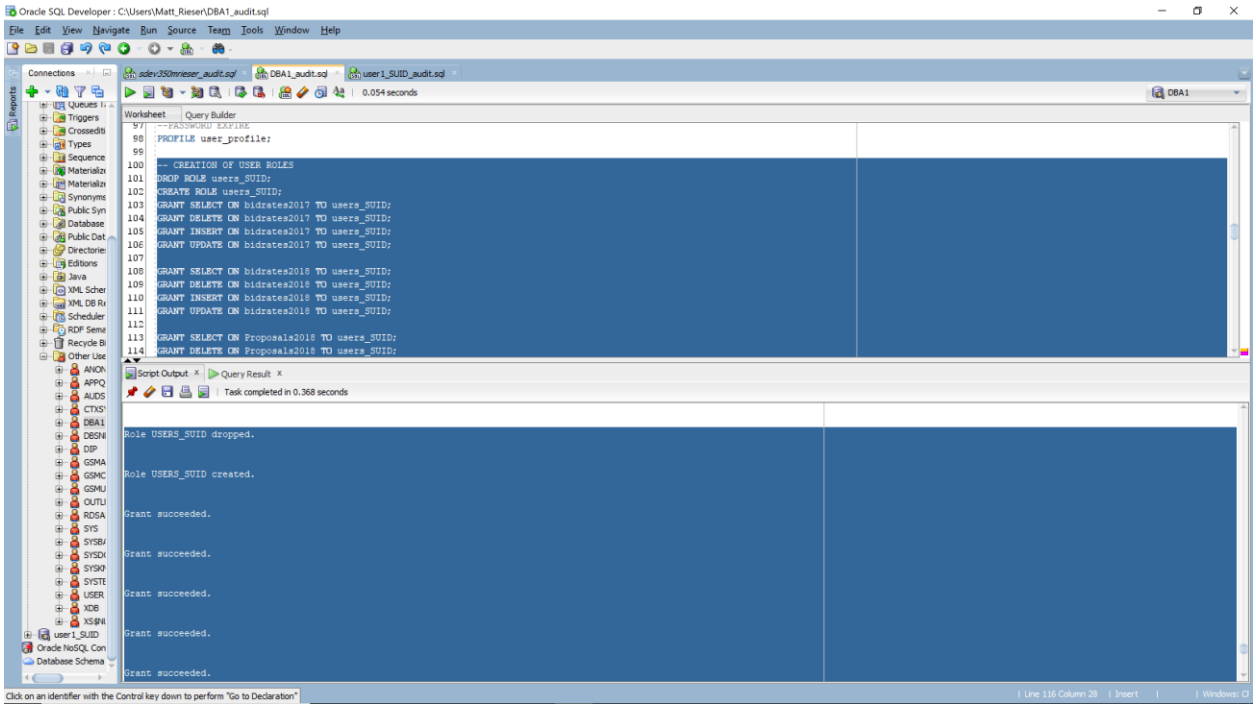


Above – shows the use of user_profile as DBA1 and user1_SUID

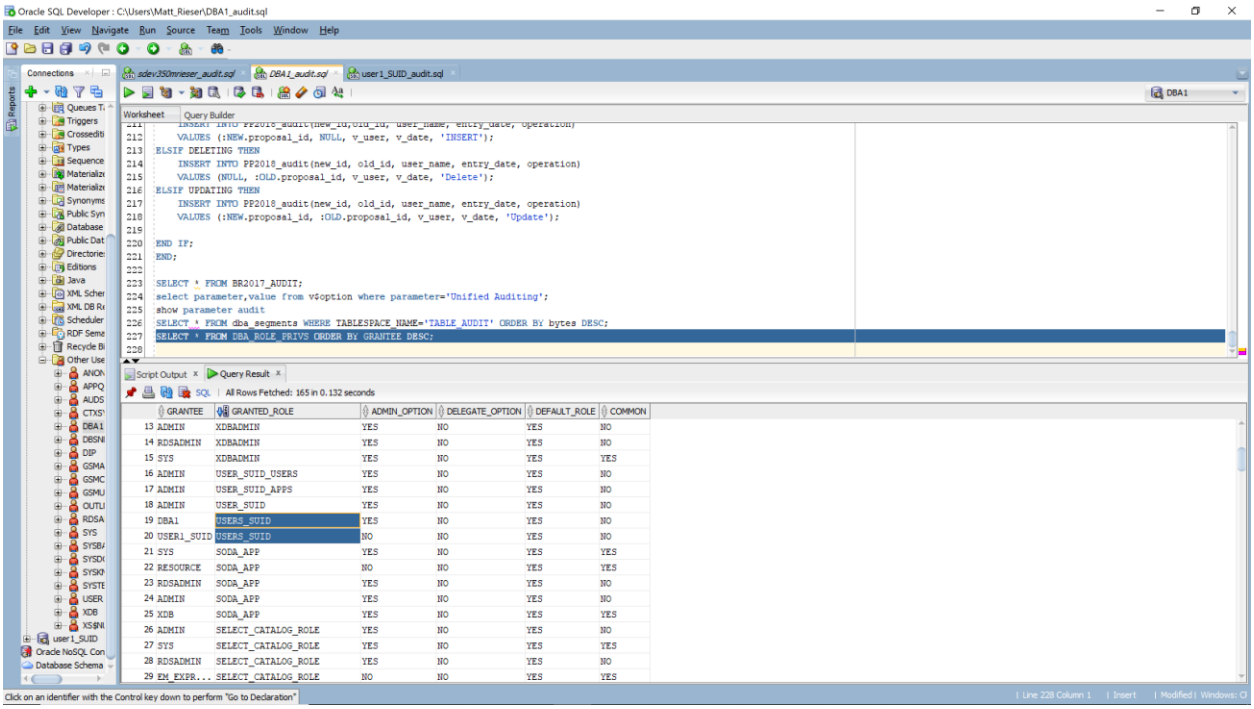
Creation and use of Roles



Above – Roles granted to DBA1, connection and 'DBA' access and session creation

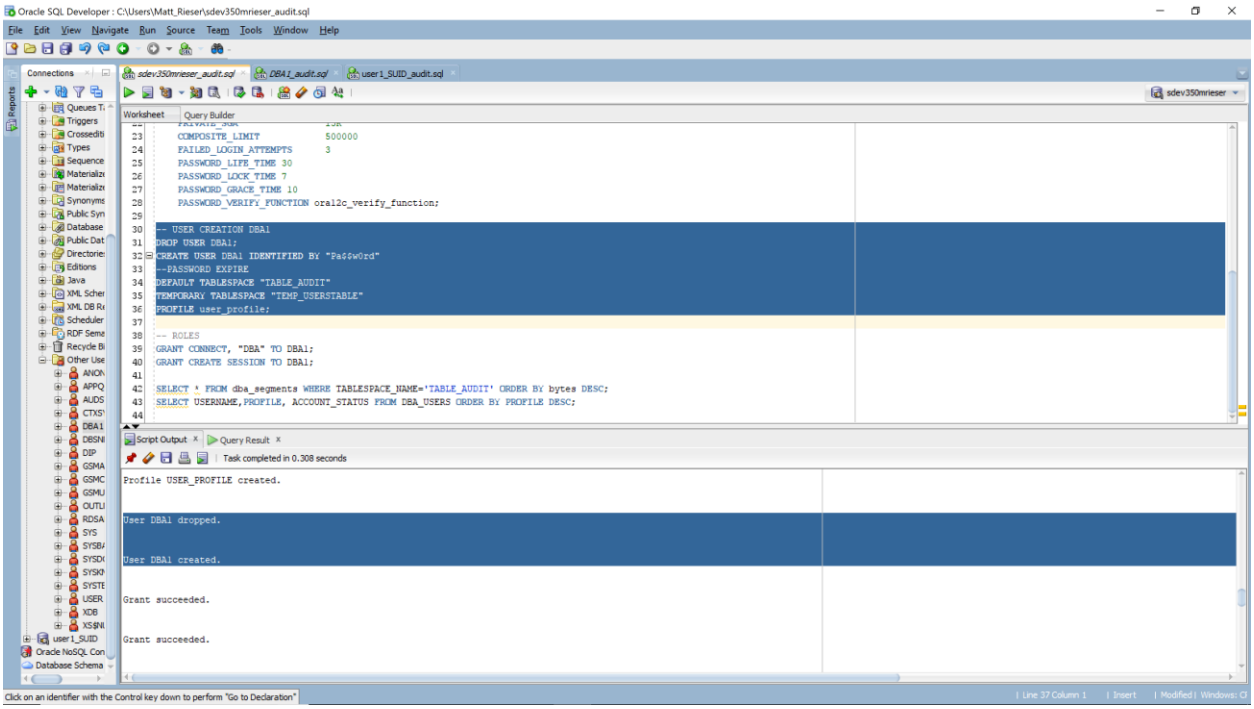


Above – Role - users_SUID in DBA1 was created and granted to user1_SUID

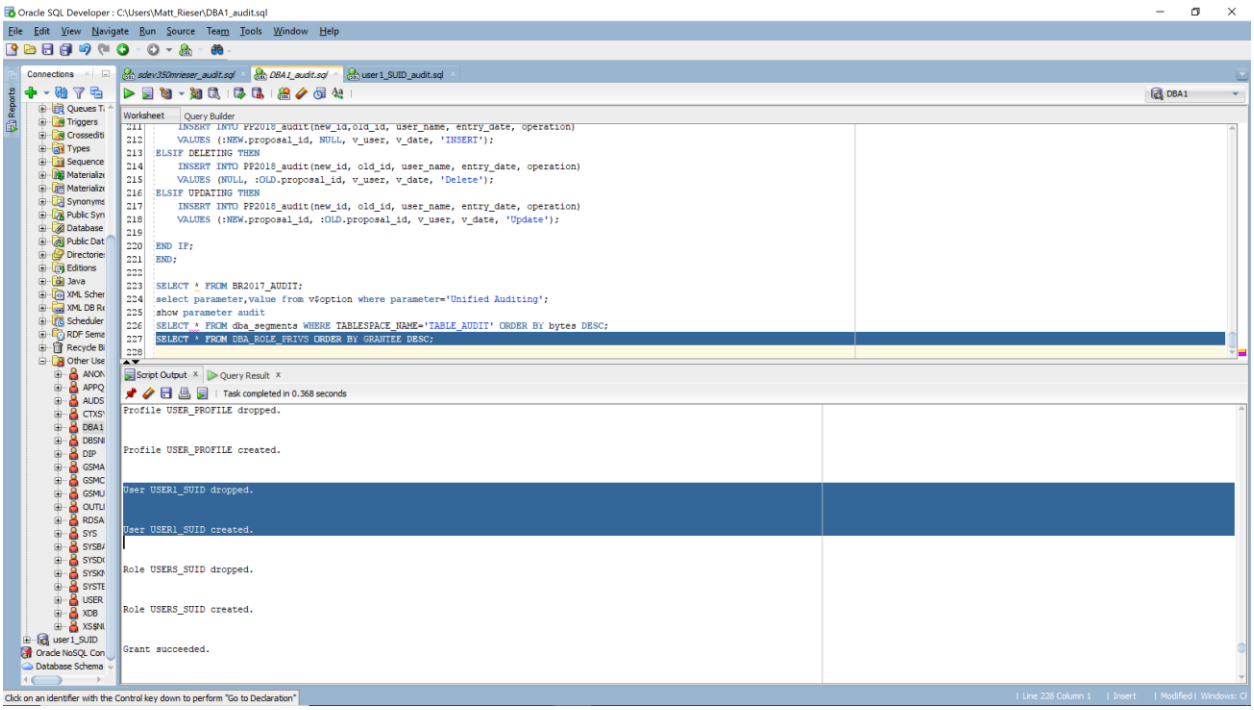


Above – we see the DBA1 and user1_SUID with their granted roles

Creation and use of Users

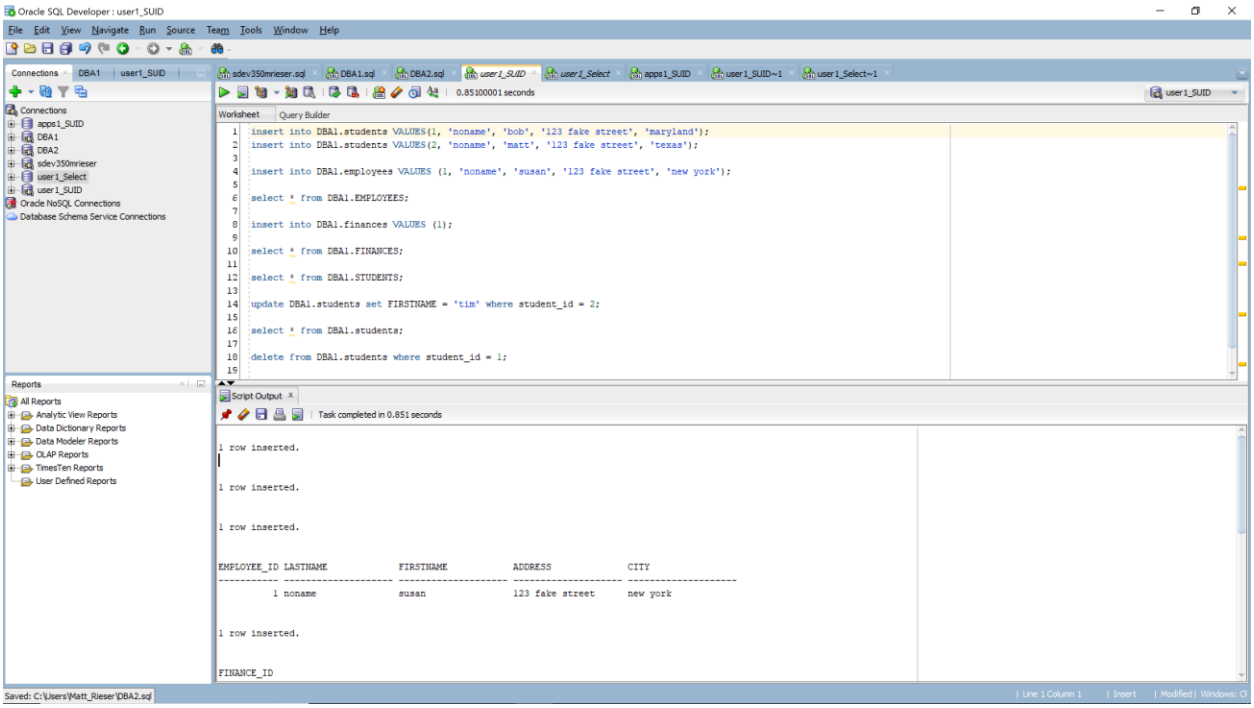


Above – successful creation of DBA user - DBA1



Above - Successful creation of 1 user with select, update, insert, and delete permissions for tables – BidRates2017, BidRates2018, Proposals2017, and Proposals2018 within DBA1

Testing DBA1 user functionality



Above – shows the successful completion of script as user1_SUID. This script -

- inserts into DBA1.BIDRATES2017, DBA1. BIDRATES2018, DBA1.PROPOSALS2017, and PROPOSALS2018 tables
- selects from DBA1.BIDRATES2017, DBA1. BIDRATES2018, DBA1.PROPOSALS2017, and PROPOSALS2018 tables
- updates record's from DBA1.BIDRATES2017, DBA1.BIDRATES2018, DBA1.PROPOSALS2017, and PROPOSALS2018 tables
- deletes a record from DBA1.BIDRATES2107
- shows that permission for table creation is not allowed

Output:

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

BR_ID BID_ID

---------- ----------

1 bob

BR_ID BID_ID

---------- ----------

1 matt

P_ID PROPOSAL_ID

---------- ----------

1 sue

P_ID PROPOSAL_ID

---------- ----------

1 lucy

1 row updated.

1 row updated.

1 row updated.

1 row updated.

BR_ID BID_ID

---------- ----------

1 MATT

BR_ID BID_ID

---------- ----------

1 MARK

P_ID PROPOSAL_ID

---------- ----------

1 DIANA

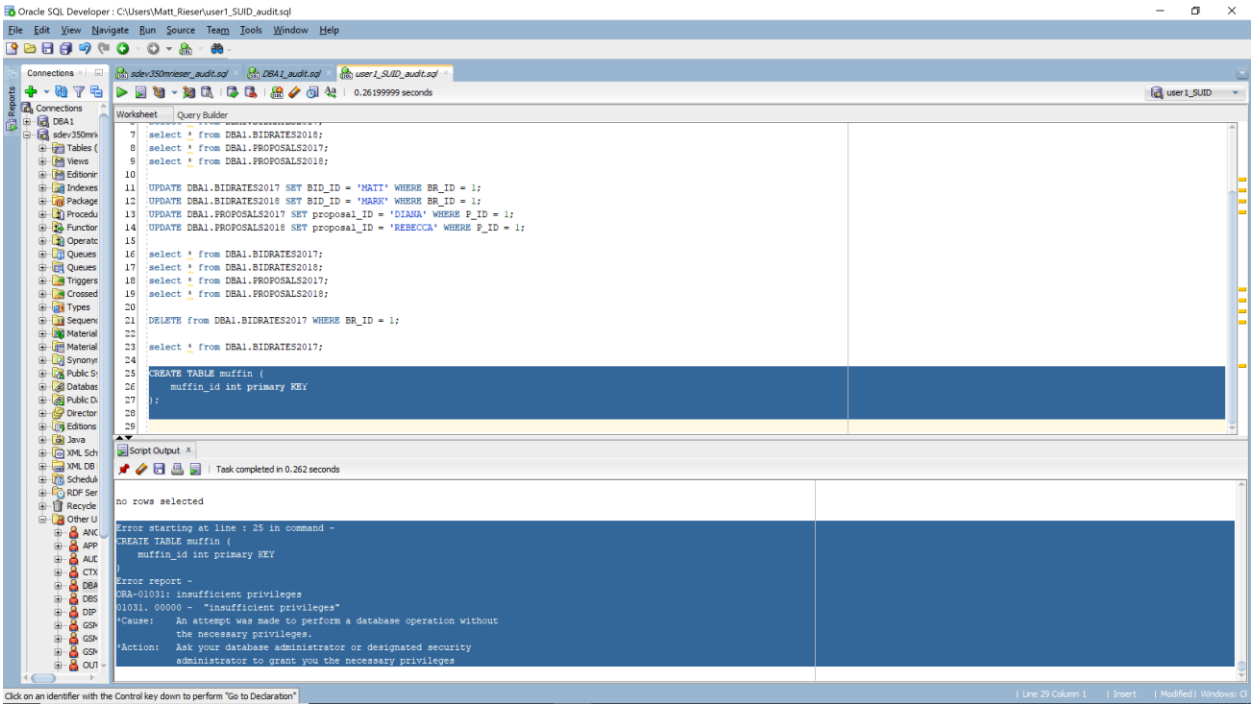
P_ID PROPOSAL_ID

---------- ----------

1 REBECCA

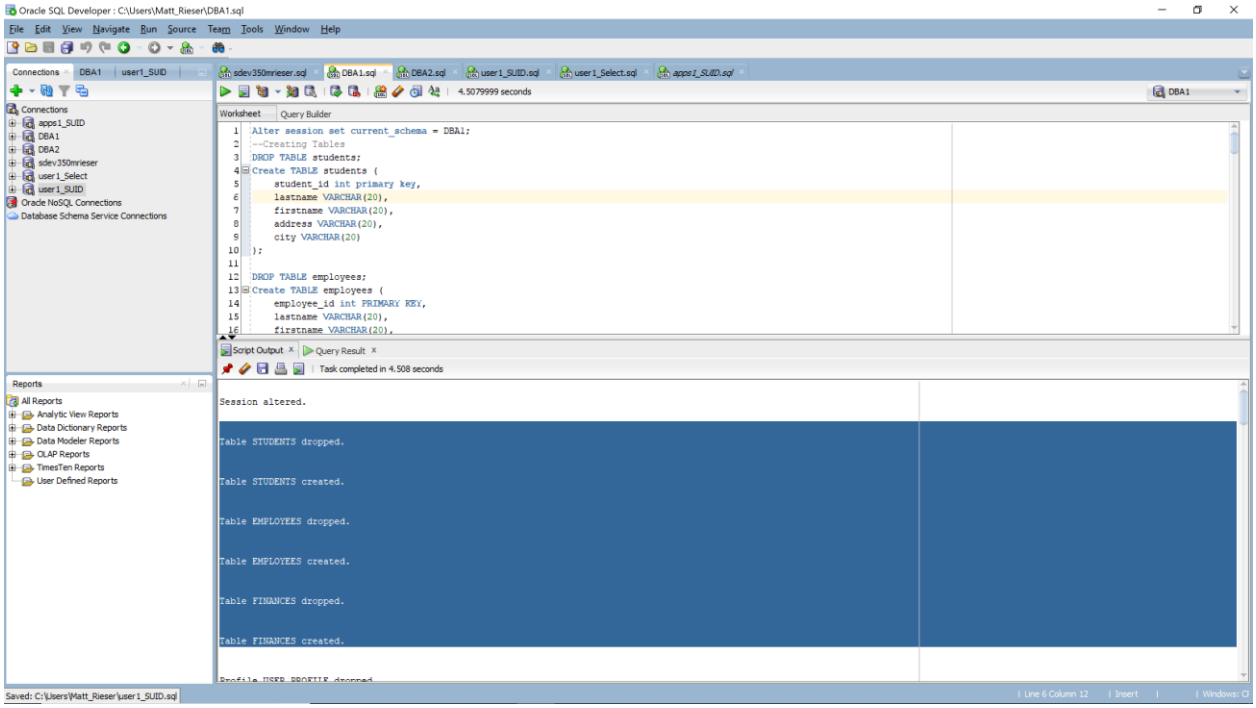
1 row deleted.

no rows selected

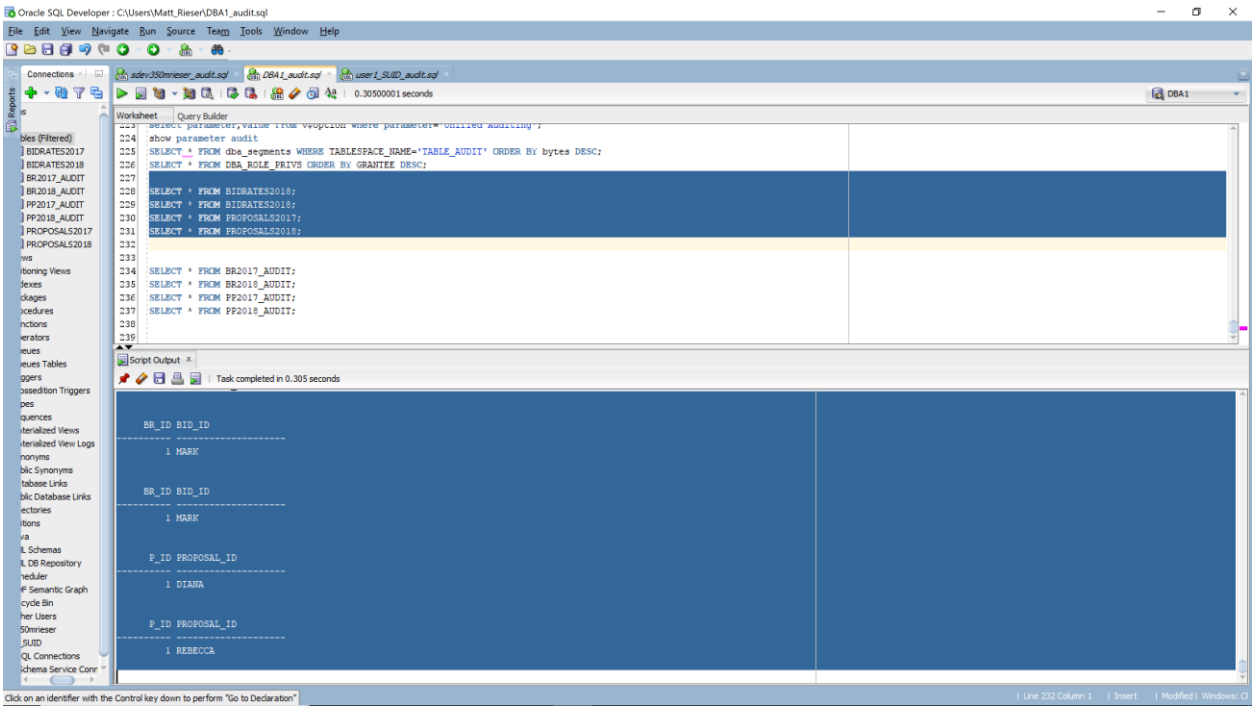


Above - tests to see if the creation of table muffin is possible, which this account doesn't have permissions for, so it outputs a permission error (good error)

Creation and use of Tables



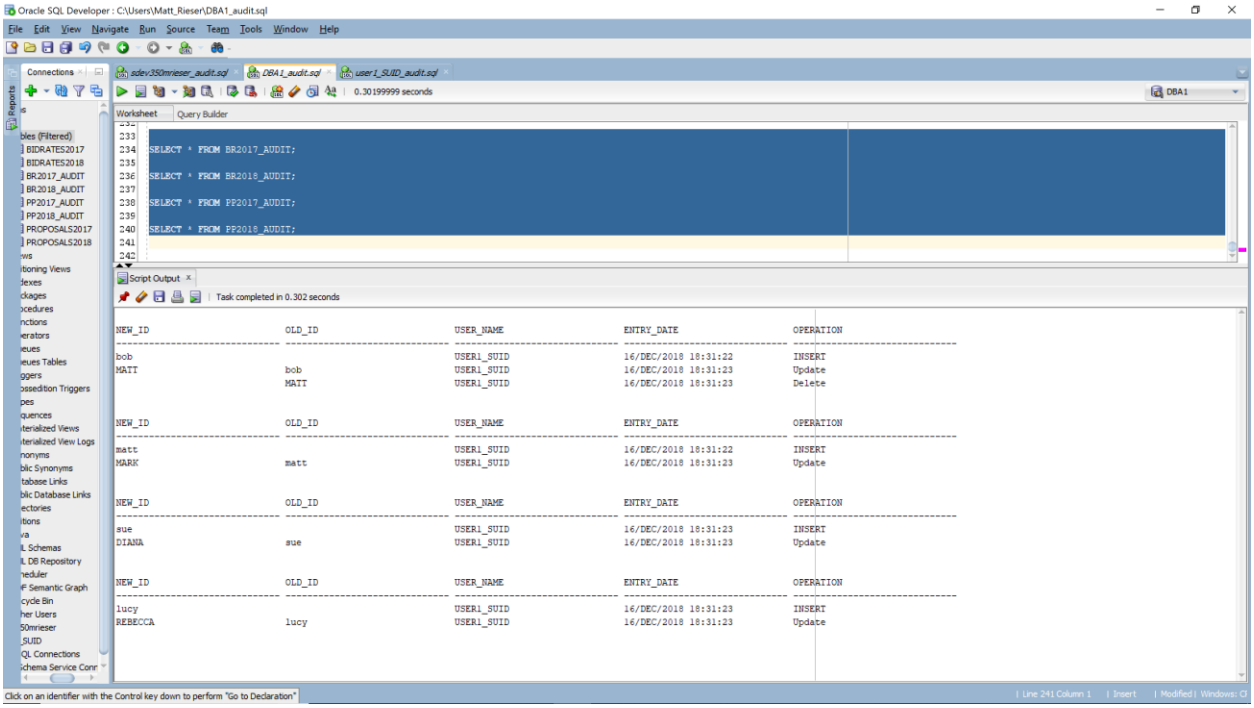
Above – shows the creations of students, employees, and finances tables within DBA1



Above – successful testing with select statements of the tables (BidRates2017, BidRates2018, Proposals2017, and Proposals2018) in DBA1 that were propagated by user1_SUID's account, previously used.

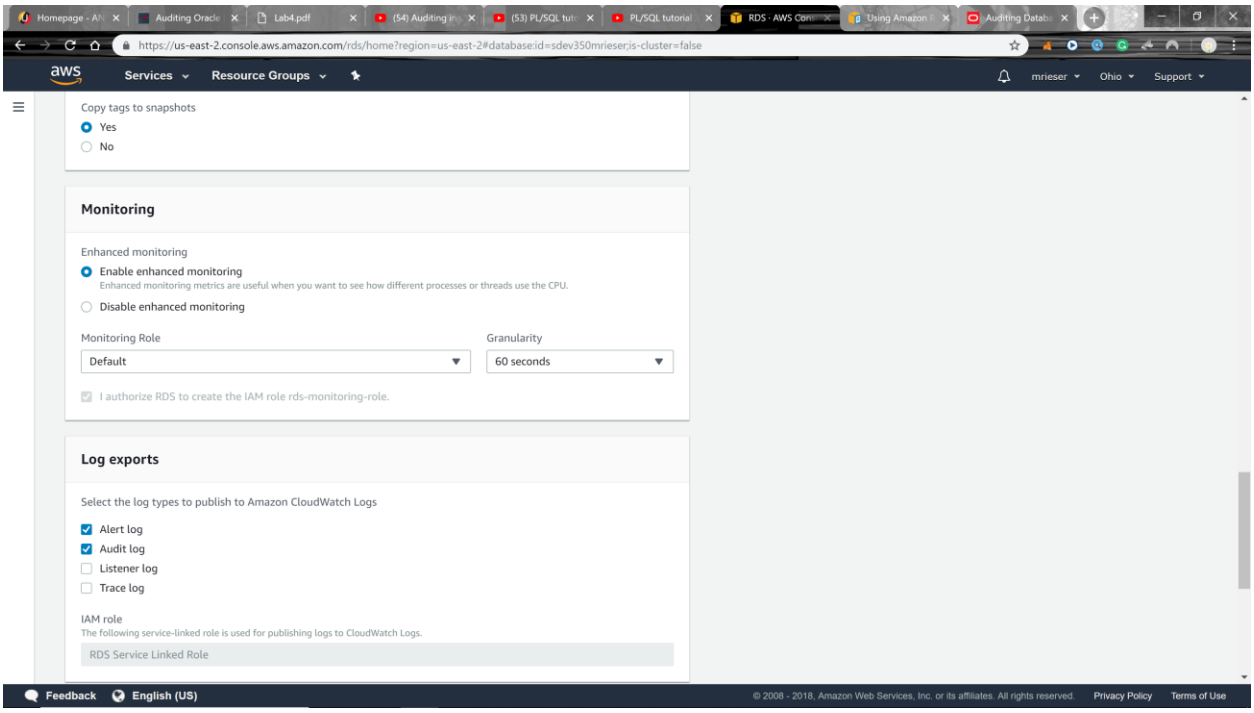
Looking at the auditing tables

After using the user1_SUID for inserting, selecting, updating, and deleting. There should be records in my auditing tables at this point (set triggers earlier to record changes within tables), showing the changes.

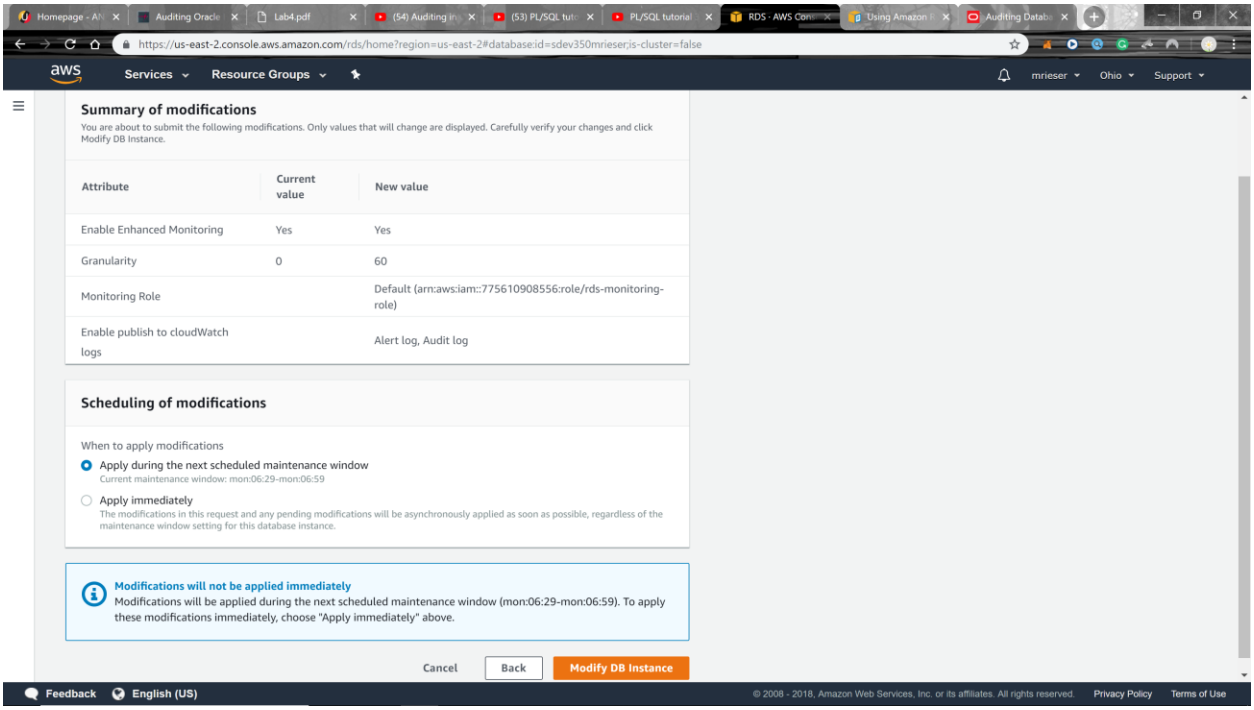


Above – successfully recorded the changes that occured within BidRates2017, BidRates2018, Proposals2017, and Proposals2018. The tables display the new_id, old_id, username, entry_date, and operation.

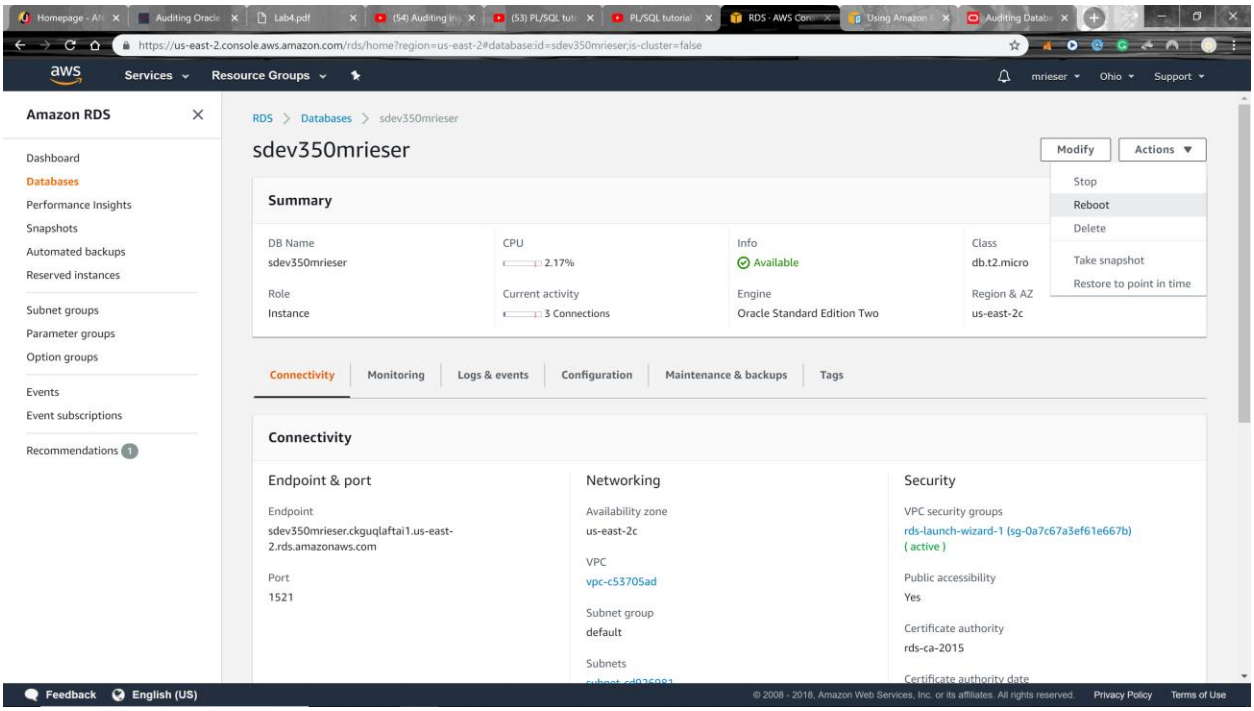
Modifying the AWS Instance to Enable Audits and Auditing



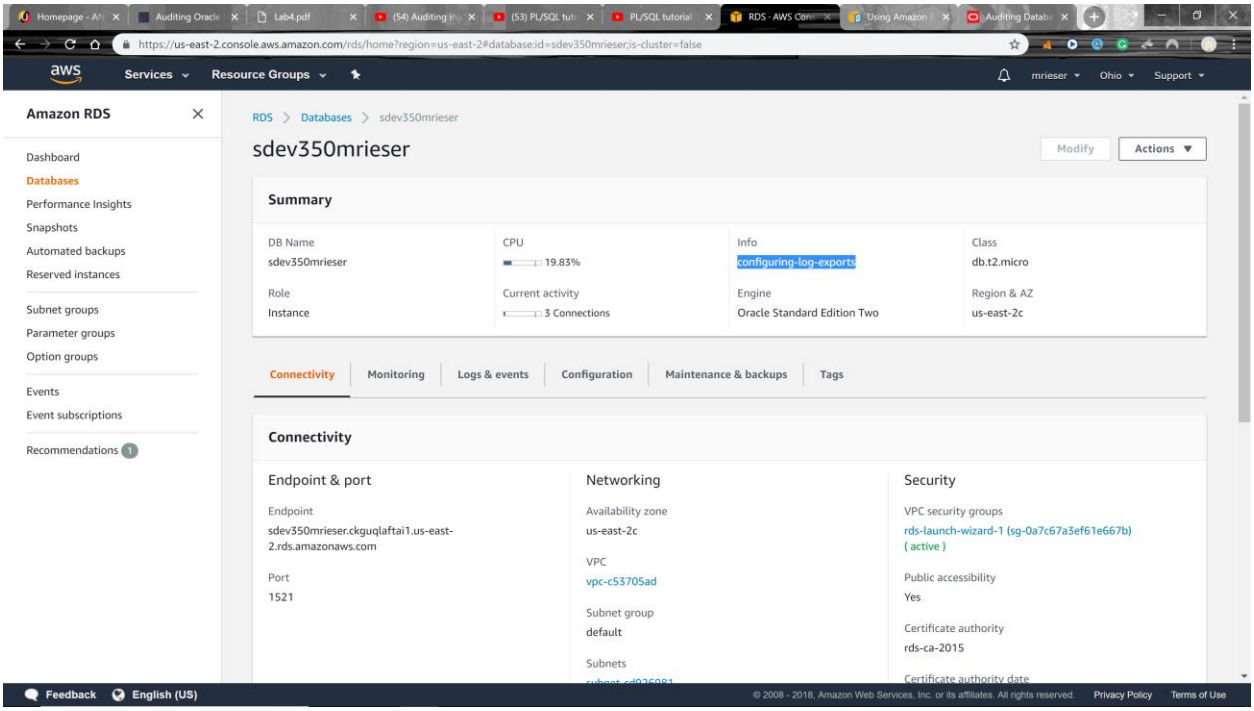
Above – clicked the enable enhanced monitoring option under Monitoring and checked boxed alert log and audit log options under Log exports.



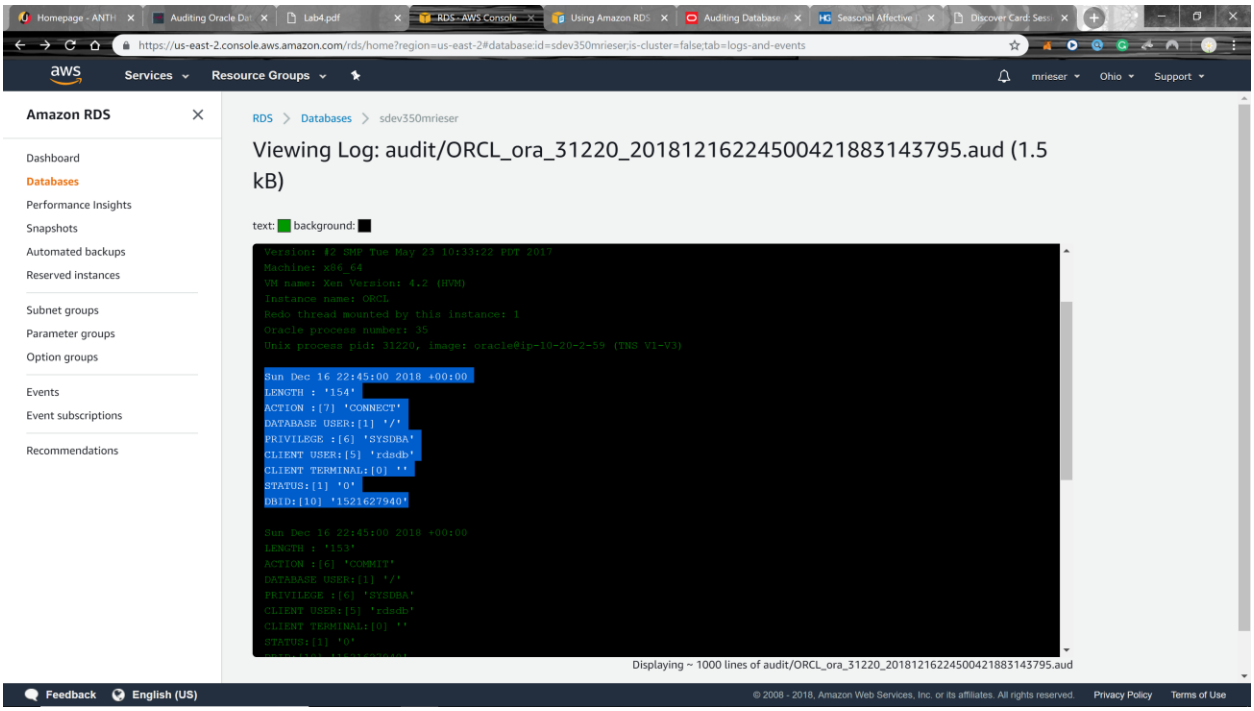
Above – clicked the modify DB instance button at the bottom of the page



Above – rebooted the instance to implement the changes

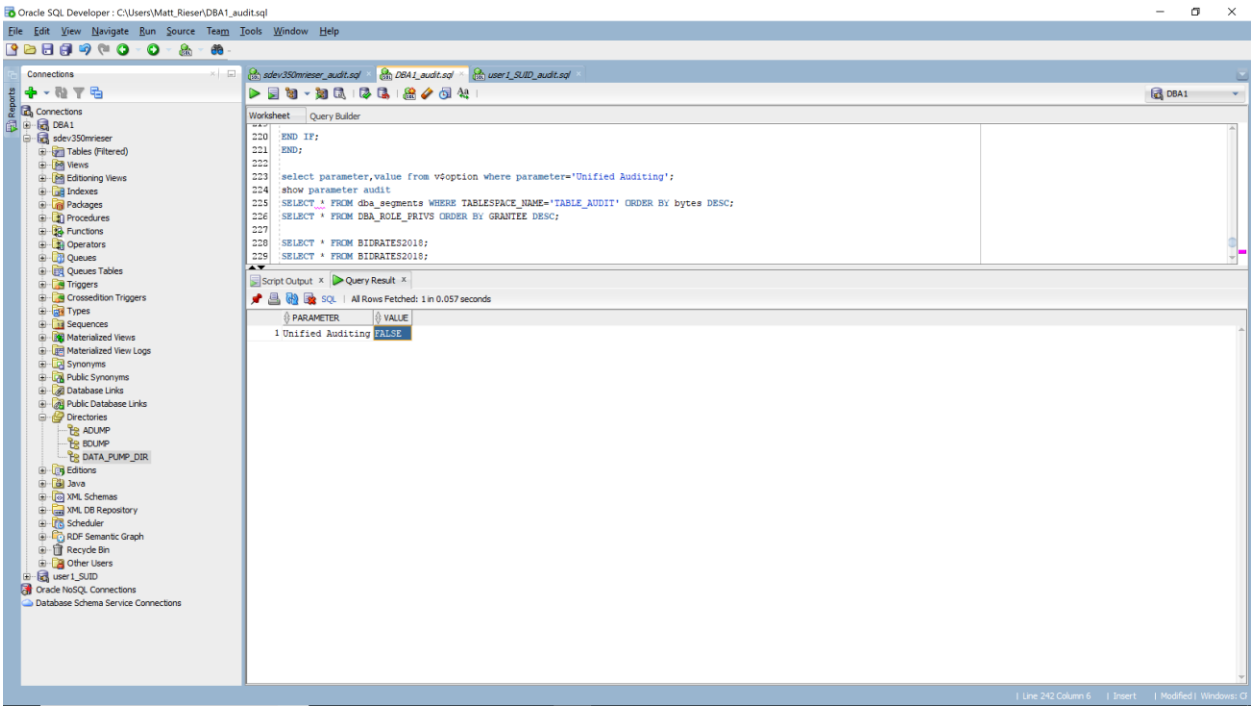


Above – instance is configuring audit changes



Above – nothing usual in the AWS logs to report on, other than commits made by me

Things that didn’t work correctly



Above – ran the command to enable unified auditing

--ENABLE UNIFIED AUDITING

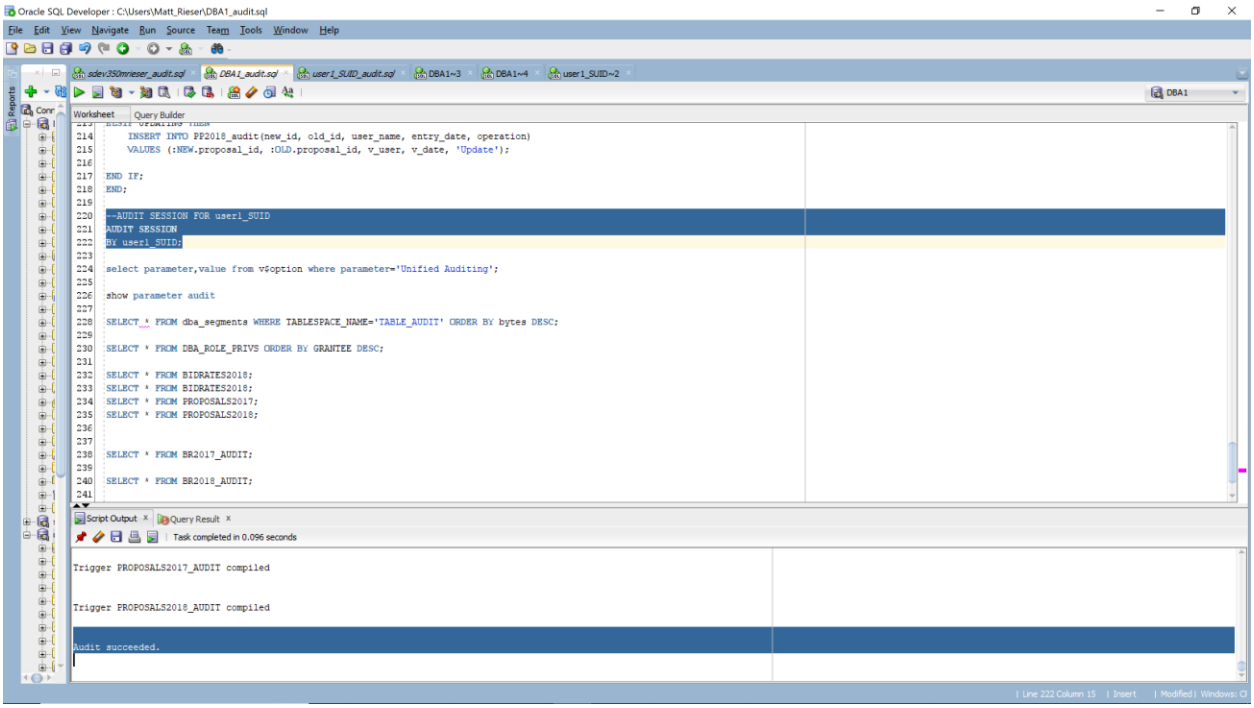
EXECUTE

DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY(DBMS_AUDIT_MGMT.AUDIT_

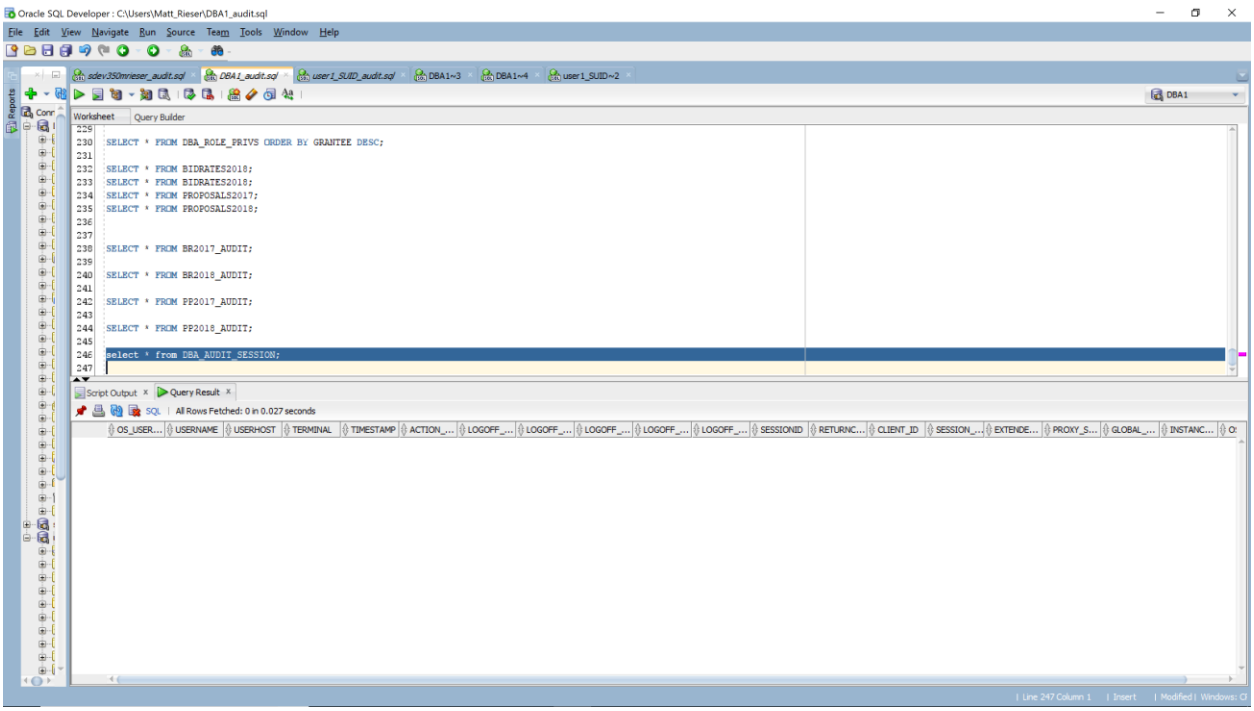
TRAIL_UNIFIED, DBMS_AUDIT_MGMT.AUDIT_TRAIL_WRITE_MODE,

DBMS_AUDIT_MGMT.AUDIT_TRAIL_IMMEDIATE_WRITE);

and then rebooted the aws instance but unified auditing stayed false. So I’m not really sure what to do.



Above – running the command AUDIT SESSION BY user1_SUID; and the audit succeeded



Above – however whenever I run the select * FROM DBA_AUDIT_SESSION; nothing shows up. The user's sessions are supposed to be displayed. Tried logging in with user1_SUID multiple times but it still wouldn't show up. If you have any theories, I'm all ears.

Miscellaneous code that I forgot to mention in the above sections

Alter session set current_schema = DBA1;

This command provides user's (examples – user1_SUID) with the ability to access the tables created, if they have been granted access to them. Example - user1_SUID (Select, Update, Insert, and Delete) can access DBA1 tables.

Examples –

*select * from DBA1.BIDRATES2017;*

Audit commands –

AUDIT SESSION

BY user1_SUID;

Used to audit user1_SUID sessions and should be displayed in DBA_AUDIT_SESSION

AUDIT SESSION

BY DBA1;

Used to audit DBA1 user's sessions and should be displayed in DBA_AUDIT_SESSION