User Defined Functions

This guide will walk you through setting up Snowflake and getting familiar with

Prerequisites

If you are not familiar with the Snowflake User Interface, then please watch the video below.

- Quick Video Introduction to Snowflake

What You'll Learn

- Snowflake account and user permissions
- Creating database objects
- Query a user-defined scalar function
- Query a user-defined table function
- Delete database objects

What You'll Need

- A Snowflake Trial Account

What You'll Build

- Database objects and user-defined functions to query those objects.

2. Begin With the Basics

First, we'll go over how to create your Snowflake account and manage user permissions.

Create a Snowflake Account

Snowflake lets you try out their services for free with a trial account. Follow the prompts to activate your account via email.

If you already have a Snowflake account, you can use it. You just need a role with permission to create a database.

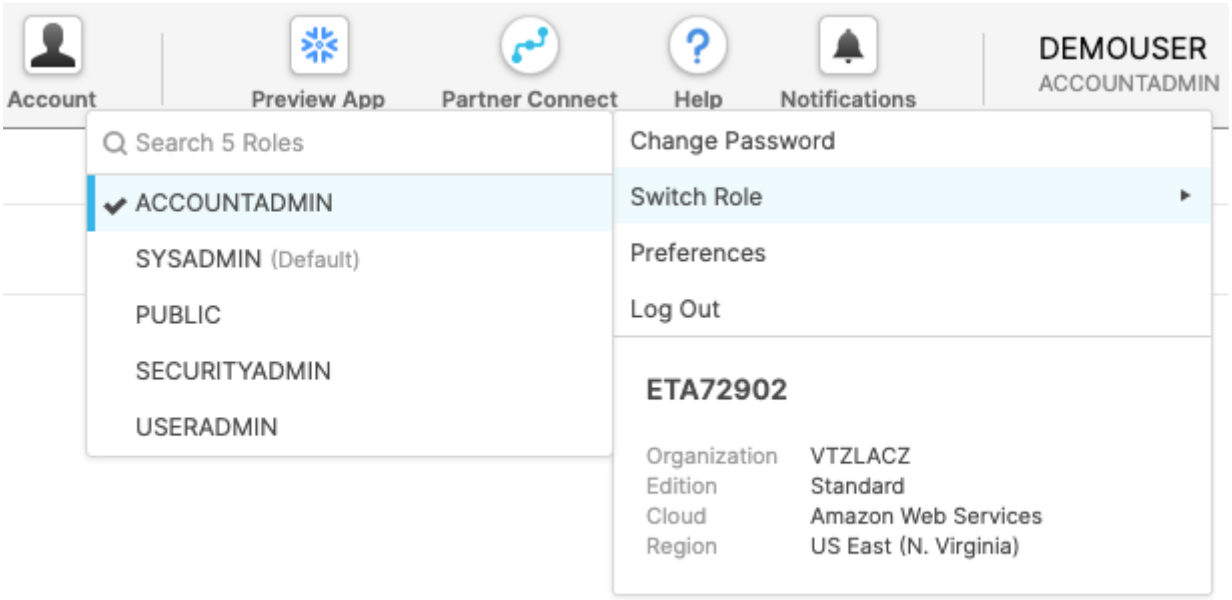
Access Snowflake's Web Console

```
https://<your-account-name>.snowflakecomputing.com/console/login
```

Log in to the web interface from your browser. The URL contains your account name and potentially the region.

Changing your role

If you already have a Snowflake account, you can use a role with privildges to create a database.



Switch the account role from the default SYSADMIN to ACCOUNTADMIN. This **is not required**, but will ensure your web UI aligns with the screen shots in this lab.

With your new account created and the role configured, you're ready to begin creating database objects in the following section.

3. Create Database Objects

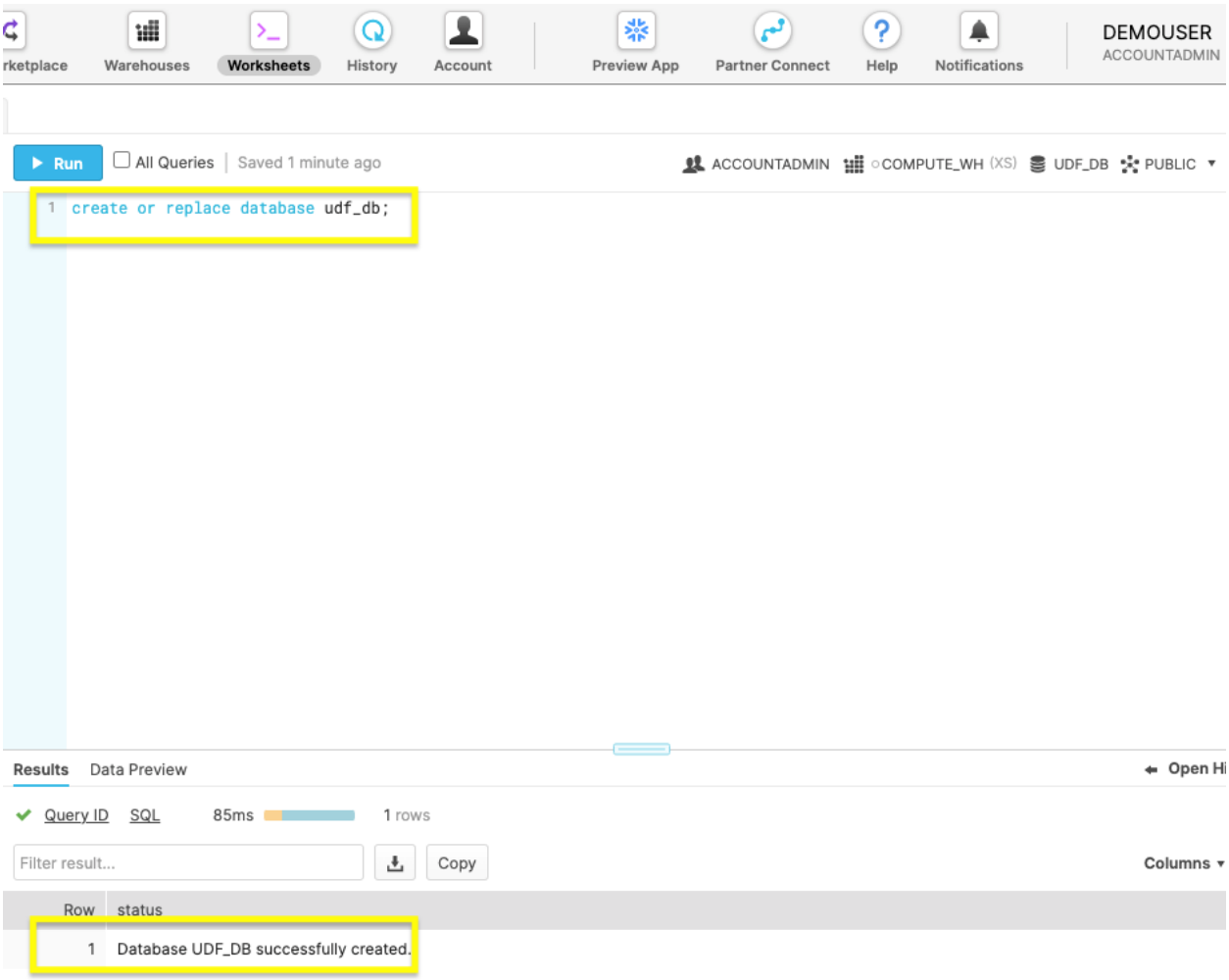
With your Snowflake account at your fingertips, it's time to create the database objects.

Within the Snowflake web console, navigate to **Worksheets** and use a fresh worksheet to run the following commands.

1. Create a new Database

```
create or replace database udf_db;
```

Build your new database named udf_db with the command above.

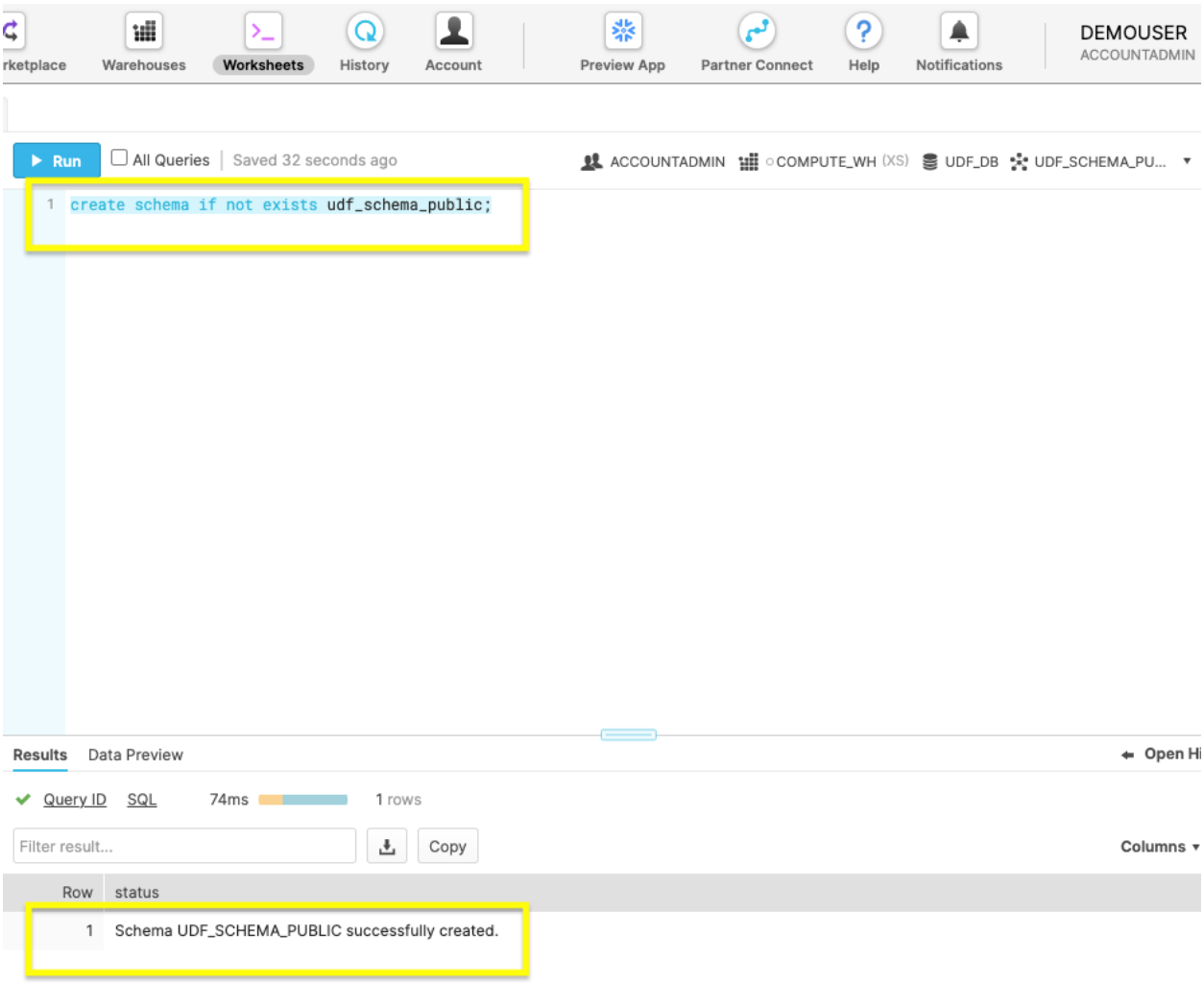


The **Results** should display a status message of **Database UDF_DB successfully created** .

2. Create a new Schema

```
create schema if not exists udf_schema_public;
```

Use the above command to whip up a schema called udf_schema_public.

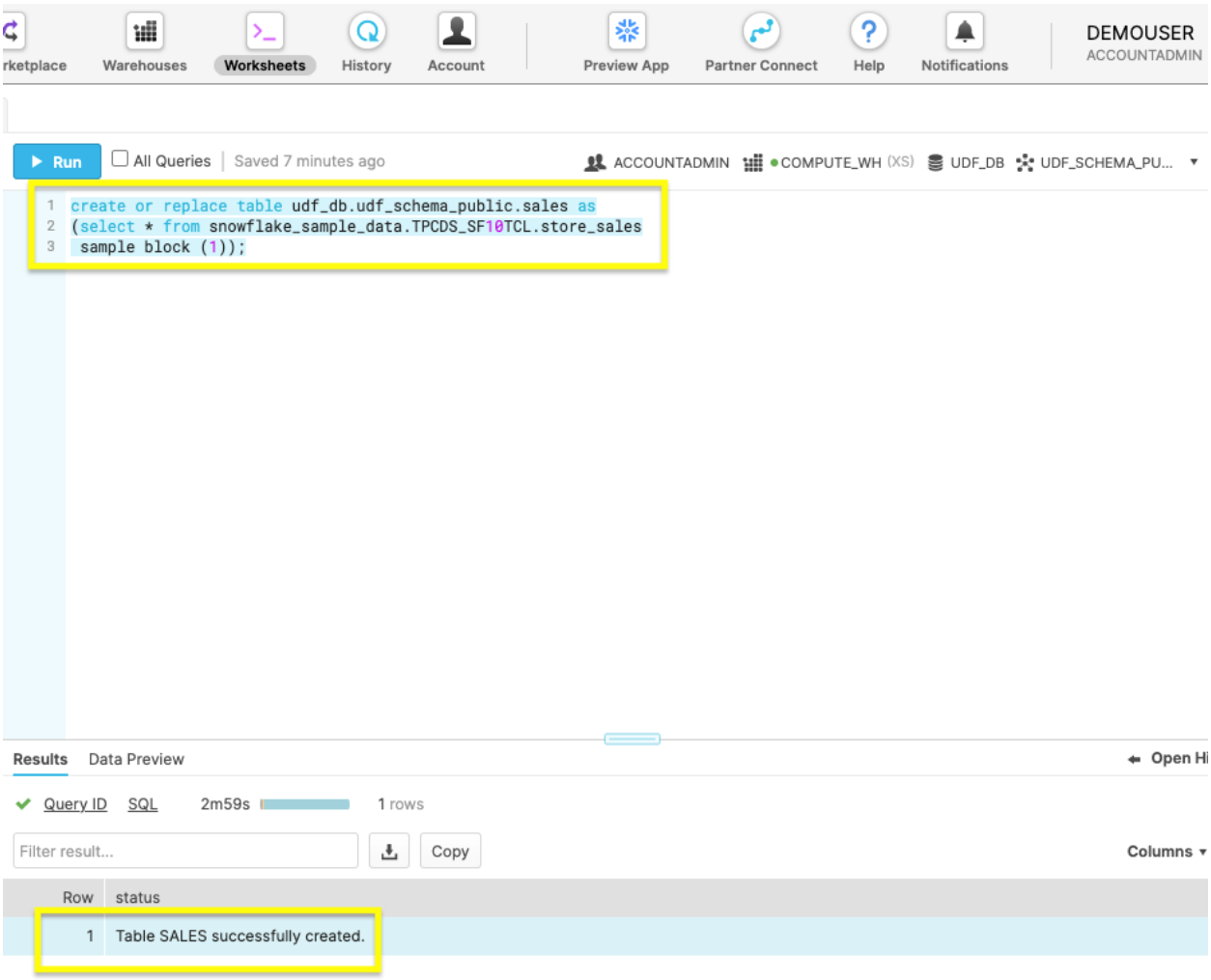


The **Results** should display a status message of **Schema UDF_SCHEMA_PUBLIC successfully created**.

3. Copy Sample Data Into New Table

```
create or replace table udf_db.udf_schema_public.sales
  as
    (select * from snowflake_sample_data.tpcds_sf10tcl.store_sales sample block (1));
```

Create a table named ‘sales' and import the sales data with this command. Bear in mind, importing the sample data will take a longer time to execute than the previous steps.



The **Results** should display a status of **Table SALES successfully created** .

With the necessary database objects created, it's time to move onto the main course of working with a UDF in the next section.

4. Execute a Scalar User-Defined Function

With the database primed with sample sales data, we're *almost* ready to try creating a scalar UDF. Before diving in, let's first understand more about UDF naming conventions.

If the function name doesn't specify the database and schema(e.x. `udf_db.udf_schema_public.udf_name`) then it defaults to the active session.

UDF

```
create function udf_max()
  returns NUMBER(7,2)
  as
    'select max(SS_LIST_PRICE) from udf_db.udf_schema_public.sales'
    ;
```

.

5. Query With User-Defined Table Function

After creating a successful scalar UDF, move onto making a function that returns a table with a UDTF(user-defined table function).

1. Create a UDTF

```
create or replace function
udf_db.udf_schema_public.get_market_basket(input_item_sk number(38))
returns table (input_item NUMBER(38,0), basket_item_sk NUMBER(38,0),
num_baskets NUMBER(38,0))
as
 'select input_item_sk, ss_item_sk basket_Item, count(distinct
ss_ticket_number) baskets
from udf_db.udf_schema_public.sales
where ss_ticket_number in (select ss_ticket_number from udf_db.udf_schema_public.sales where
ss_item_sk = input_item_sk)
group by ss_item_sk
order by 3 desc, 2';
```

We've covered a lot of ground! Before we wrap-up, drop the practice database you created in this guide. This will remove the database and all of the tables and functions that you created.

Drop the Database

Drop the database: udf_db.

```
drop database if exists udf_db;
```

Verify the database is entirely gone by checking the Results for UDF_DB successfully dropped.

You should now have a good handle on SQL UDFs by practicing both scalar and table functions. With our database objects cleared, it's time to look ahead.

Consider the potential of a sharable and secure user-defined function. You can learn how to share user-defined functions, such as the market basket analysis table, following this post about the power of secure UDFs.

What we've covered

- Registered a Snowflake account
- Configured role permissions
- Created a database and other objects
- Created a table to analyze data with a UDTF
- Queried a custom UDF
- Droping a database to clean up all objects