CSE 414 Homework 3: Advanced SQL and Azure

Objectives: To practice advanced SQL. To get familiar with commercial database management systems (SQL Server) and using a database management system in the cloud (SQL Azure).

Assignment tools: SQL Server on Windows Azure through SQL Azure. SQL Server Management Studio has been installed on the CSE lab and VDI machines if you would like to use that instead of Azure's web interface.

Assigned date: Thursday, Oct 15, 2020

Due date: Tuesday, Oct 27, 2020

What to turn in: hw3-q1.sql, hw3-q2.sql, etc. (see below).

Assignment Details

This homework is a continuation of homework 2 but with three changes:

- The queries are more challenging
- You will get to use a commercial database system (i.e., no more SQLite :). SQLite simply cannot execute these queries in any reasonable amount of time; hence, we will use SQL Server, which has one of the most advanced query optimizers.
- You will use the Microsoft Azure cloud.

In this homework, you will do three things:

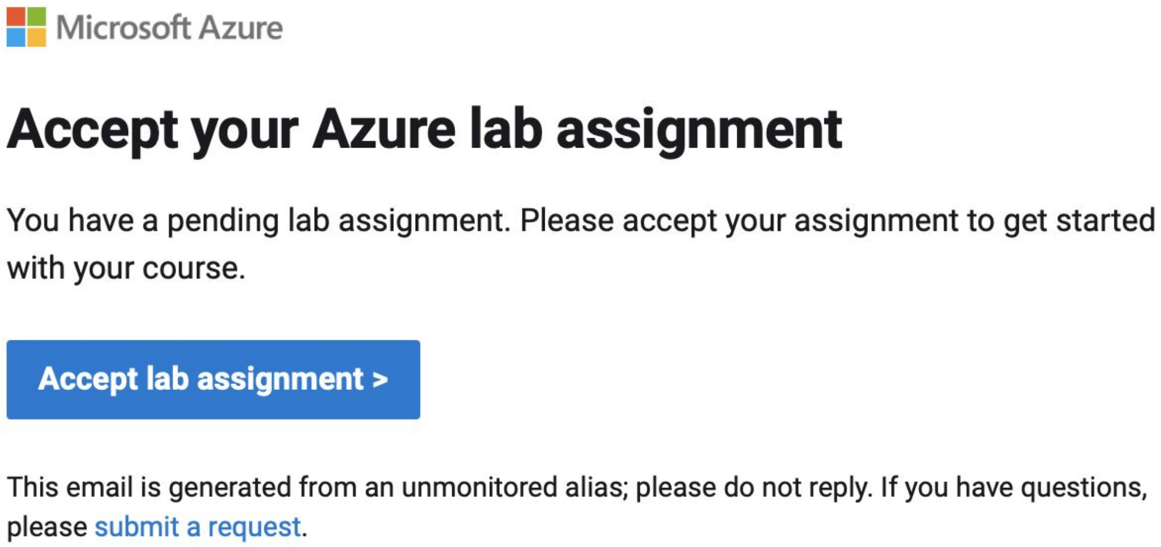
1. Create a database in the SQL Server database management system running as a service on Windows Azure; import data from an Azure public blob.
2. Write and test the SQL queries below; keep in mind that the queries are quite challenging, both for you and for the database engine.
3. Reflect on using a database management system running in a public cloud.

A. Setting up an Azure SQL Database [0 points]

In this assignment, we want you to learn how to use an Azure SQL database from scratch. Your first step will thus be to setup a database in the Azure service and importing your data. This step may seem tedious but it is crucially important. We want you to be able to continue using Azure after the class ends. For this, you need to know how to use the system starting from nothing.

NOTE: These steps will take some time to complete, so start early!

Step 1: Create an Azure account and log in to Azure portal



Microsoft Azure

Accept your Azure lab assignment

You have a pending lab assignment. Please accept your assignment to get started with your course.

Accept lab assignment >

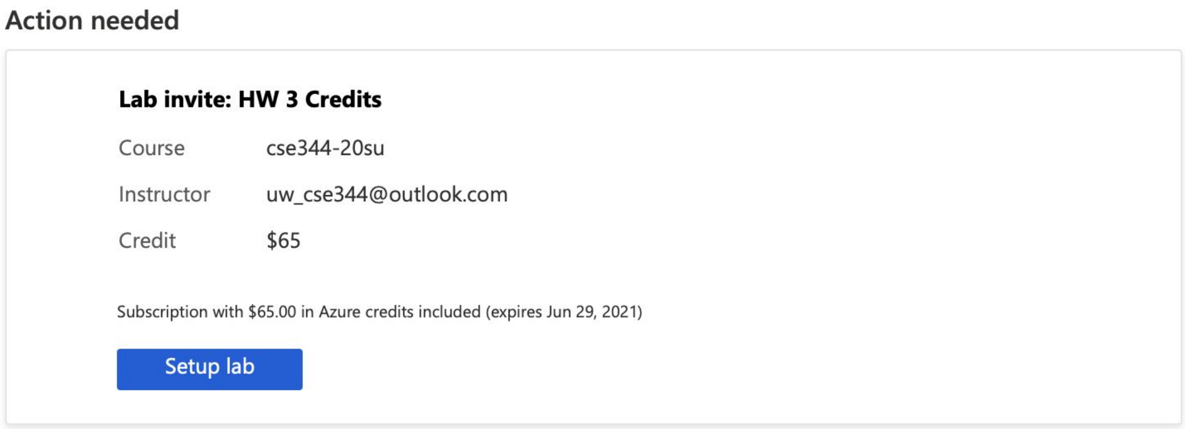
This email is generated from an unmonitored alias; please do not reply. If you have questions, please submit a request.

Click on the "Accept lab assignment" link in the email "Action required: Accept your lab assignment", log in using your uw.edu account and password.

Afterward, you will be forwarded to the Azure portal.

Under “Action needed”, click “Setup lab”. This might take a while.

- Note: Lab invite name might vary, correct names include “HW 3 Credits” and “Microsoft sponsorship 2”



Action needed

Lab invite: HW 3 Credits

Course	cse344-20su
Instructor	uw_cse344@outlook.com
Credit	$65

Subscription with $65.00 in Azure credits included (expires Jun 29, 2021)

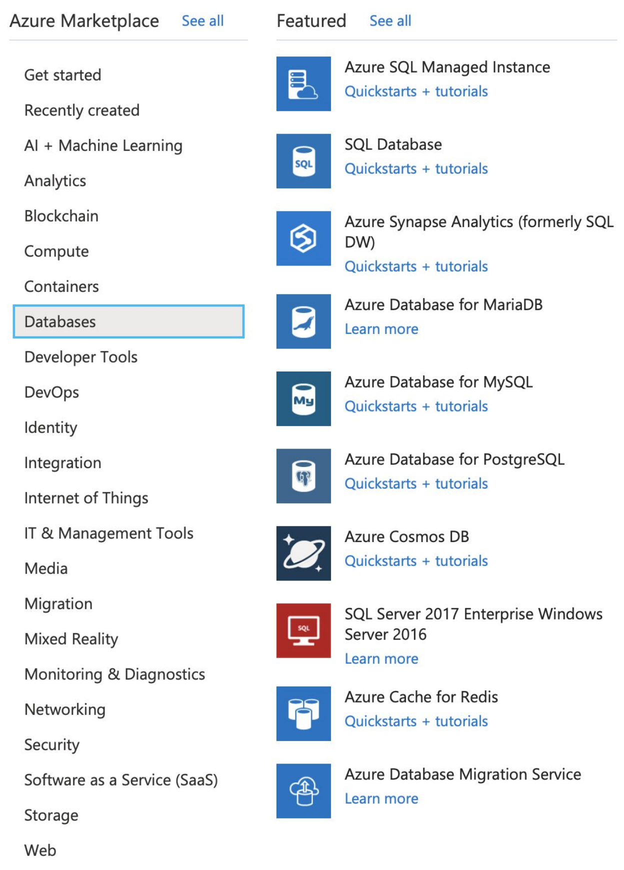
Setup lab

Step 2: Learn about Azure SQL Server

Spend some time clicking around, reading documentation, watching tutorials, and generally familiarizing yourself with Azure and SQL Server.

Step 3: Create a database

From the [Azure portal](), select "+ (Create a resource)" on the left, + then select "Databases", then select "SQL Database". This will bring up a panel with configuration options for a new DB instance.



Azure Marketplace See all

Get started
Recently created
AI + Machine Learning
Analytics
Blockchain
Compute
Containers
Databases
Developer Tools
DevOps
Identity
Integration
Internet of Things
IT & Management Tools
Media
Migration
Mixed Reality
Monitoring & Diagnostics
Networking
Security
Software as a Service (SaaS)
Storage
Web

Featured See all

Azure SQL Managed Instance
Quickstarts + tutorials

SQL Database
Quickstarts + tutorials

Azure Synapse Analytics (formerly SQL DW)
Quickstarts + tutorials

Azure Database for MariaDB
Learn more

Azure Database for MySQL
Quickstarts + tutorials

Azure Database for PostgreSQL
Quickstarts + tutorials

Azure Cosmos DB
Quickstarts + tutorials

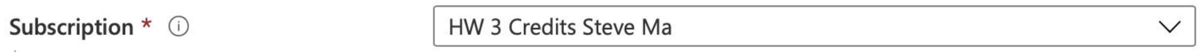
SQL Server 2017 Enterprise Windows Server 2016
Learn more

Azure Cache for Redis
Quickstarts + tutorials

Azure Database Migration Service
Learn more

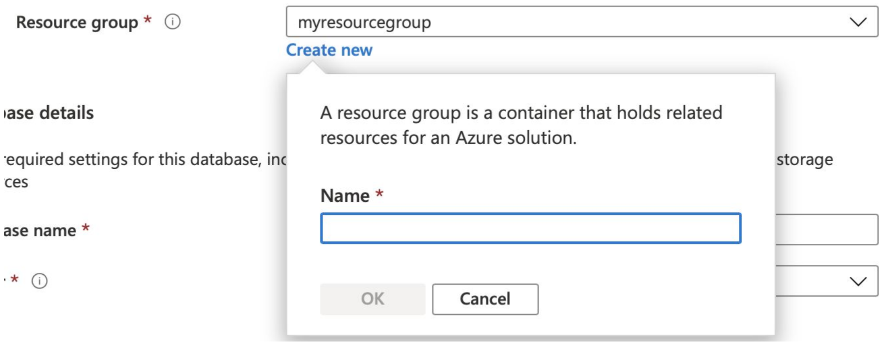
Perform the following configuration:

- Make sure your subscription is not set to "Free Trial" or "Azure for Students" (This may cause your database to expire in the middle of the quarter when you need it for future assignments)
    - Note: Subscription name might vary, correct subscription names include “HW 3 Credits” and “Microsoft sponsorship 2”.



Subscription * HW 3 Credits Steve Ma

- Create a new resource group with a name (e.g., "myresourcegroup").



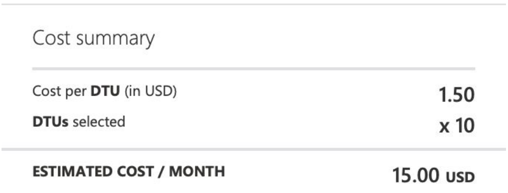
Resource group * myresourcegroup

Create new

A resource group is a container that holds related resources for an Azure solution.

Name *

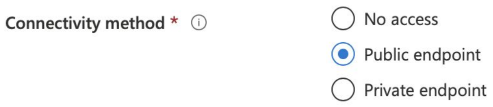
- Choose a database name (e.g., "cse414-20au").
- Create a new server by clicking on "Server". A second panel will appear to the right. Fill in the form as follows:
  - Choose a name for the server (e.g., "foobarsqlserver"). Unlike your database name, the server name must be unique across the universe of Azure SQL databases.
  - Choose an admin login and password. (We suggest you choose a password that you do not use for other accounts. You will need this when accessing your database with tools other than the portal.)
  - Set the location based on where you are at..
  - Click "OK".
- Make sure "Want to use SQL elastic pool?" is set to "No".
- Under "Compute + storage", click "Configure database". A second panel will open to the right. On this form,
  - Click "Looking for basic, standard, premium?"
  - Select "**Standard**".
  - Check that DTUs are set to **10**, max data size to **250 GB** (this is the current default setting). It should now say the monthly cost is only **$15/month**.



Cost summary

Cost per **DTU** (in USD)	1.50
DTUs selected	x 10
ESTIMATED COST / MONTH	15.00 USD

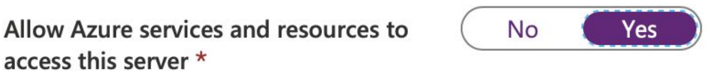
  - Click "Apply".
- Select "Next: Networking", which brings you to another panel.
  - Select "Connectivity method" : Public endpoint



Connectivity method * ⓘ

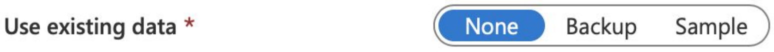
○ No access
● Public endpoint
○ Private endpoint

  - Under Firewall rules, make sure "Allow azure services to access server" is set to "yes".



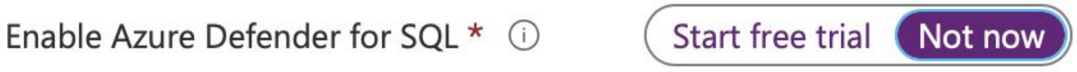
Allow Azure services and resources to access this server * No [✓ Yes]

- Click "Next: Additional settings"
  - Make sure that "Use existing data" is set to "None".



Use existing data * [✓ None] Backup Sample

  - Make sure "Azure Defender for SQL" is set to "Not now".

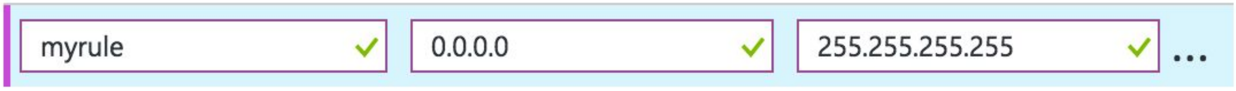


Enable Azure Defender for SQL * ⓘ Start free trial [✓ Not now]

- Click "Review + create"
- Double-check that your settings are correct, and click "Create". This may take a few minutes to deploy.

- Once it's created, select your new database. Select the pushpin icon to "Pin to dashboard" so you can easily find it in the future.

Finally, click "Set server firewall" at the top of the page. You may need to change the settings before you can upload data. The easiest option is to add a rule that allows connections from any client, which you can do as follows:



myrule ✓ 0.0.0.0 ✓ 255.255.255.255 ✓ ...

Be sure to click "Save" once you have added this rule.

Step 4: Try out the database

The simplest way to play with the database is using the built-in Query editor in the Azure portal. To launch this, go back to the dashboard, then click on the SQL database that you just setted. Enter the editor by clicking the "Query editor (preview)" on the side bar.

Enter the username and password that you chose when you created your database in Step 3. Once you have done that, you can try entering SQL commands. Press the "Run" button to execute them.

Now you are ready to move on to the next part of the assignment!

B. Ingesting Data (0 points)

Next, you will import all the data from HW2. The data used in this assignment is the same as that in hw2, in the flights-small.csv, carriers.csv, months.csv, and weekdays.csv files.

First, execute your CREATE TABLE statements so that the tables you will add tuples to already exist. You should paste these CREATE TABLE statements into the query editor and run them. Also, make sure that the types of the columns in the tables you created match the data, which you can see from the left sidebar. The ".import" statements should be omitted, as they are SQLite specific. We will do the import step next.

The easy, recommended way to ingest the data is by importing the data from a public blob storage container that we created for you. A "public blob" is what Microsoft calls its shareable storage hosted in the Azure cloud. Often it is easier to import data from within the same cloud (Azure), as opposed to data from outside the cloud (e.g., your local computer).

1. **Create Blob**: Run the following query, which creates a link between the name cse414blob and our existing data repo for the flights data. You only need to run this once:

```
CREATE EXTERNAL DATA SOURCE cse414blob
WITH (  TYPE = BLOB_STORAGE,
      LOCATION = 'https://cse414blob.blob.core.windows.net/cse414blob'
);
```

2. **Import data**: Run the following queries, which run a bulk import to fill the tables from the csv files in the public blob. Run each “bulk insert” statement individually. If errors occur, see if the table actually has tuples by testing “SELECT COUNT(*) FROM <TABLE>”. If the import failed, recreate the table and try the import again. Flights should be the last table you import because of the foreign keys.:

```
bulk insert Carriers from 'carriers.csv'
with (ROWTERMINATOR = '0x0a',
DATA_SOURCE = 'cse414blob', FORMAT='CSV', CODEPAGE = 65001, --UTF-8 encoding
FIRSTROW=1,TABLOCK);

bulk insert Months from 'months.csv'
with (ROWTERMINATOR = '0x0a',
DATA_SOURCE = 'cse414blob', FORMAT='CSV', CODEPAGE = 65001, --UTF-8 encoding
FIRSTROW=1,TABLOCK);

bulk insert Weekdays from 'weekdays.csv'
with (ROWTERMINATOR = '0x0a',
DATA_SOURCE = 'cse414blob', FORMAT='CSV', CODEPAGE = 65001, --UTF-8 encoding
FIRSTROW=1,TABLOCK);

-- Import for the large Flights table
-- This last import might take a little under 10 minutes on the provided server settings

bulk insert Flights from 'flights-small.csv'
with (ROWTERMINATOR = '0x0a',
DATA_SOURCE = 'cse414blob', FORMAT='CSV', CODEPAGE = 65001, --UTF-8 encoding
FIRSTROW=1,TABLOCK);
```

Do some SELECT count(*) statements to check whether your imports were successful.

- Carriers has **1594** rows
- Months has **12** rows
- Weekdays has **8** rows
- Flights has **1148675** rows

C. SQL Queries (90 points):

For each question below, write a single SQL query to answer that question and save your submission in individual files hw3-q1.sql, hw3-q2.sql, etc. Each .sql file should include:

- The SQL query that once executed returns the expected result.
- A comment that indicated the number of rows your query returns
- A comment that indicates how long the query took, and
- A comment that contains the first 20 rows of the result (if the result has fewer than 20 rows, output all of them).
    - You can simply copy and paste the first rows into the comment.

Note that SQL Server interprets NULL values differently than sqlite! Try using it in a WHERE predicate and you will see the difference.

Now answer the following questions:

1. For each origin city, find the destination city (or cities) with the longest direct flight. By direct flight, we mean a flight with no intermediate stops. Judge the longest flight in time, not distance. (15 points)
   Name the output columns **origin_city**, **dest_city**, and **time** representing the flight time between them. Do not include duplicates of the same origin/destination city pair. Order the result by origin_city and then dest_city (ascending, i.e. alphabetically).
   [Output relation cardinality: 334 rows]

2. Find all origin cities that only serve flights shorter than 3 hours. You should not include cancelled flights in your determination. (15 points)
   Name the output column **city** and sort them in ascending order alphabetically. List each city only once in the result.
   [Output relation cardinality: 109]

3. For each origin city, find the percentage of departing flights shorter than 3 hours.You should not include cancelled flights in your determination. (15 points)
   Name the output columns **origin_city** and **percentage**. Order by percentage value, then city, ascending. Be careful to handle cities without any flights shorter than 3 hours. You should return 0 as the result for these cities, not NULL (which is shown as a blank cell in Azure) Report percentages as percentages not decimals (e.g., report 75.25 rather than 0.752534).
   [Output relation cardinality: 327]

4. List all cities that can be reached from Seattle through one stop (i.e., with any two flights that go through an intermediate city) but **cannot** be reached through a direct flight. **Do not include Seattle as one of these destinations (even though you could get back with two flights)**. (15 points)
   Name the output column **city**. Order the output ascending by city.
   [Output relation cardinality: 256]

5. List all cities that cannot be reached from Seattle through a direct flight nor with one stop (i.e., with any two flights that go through an intermediate city). Warning: this query might take a while to execute. We will learn about how to speed this up in lecture. (15 points)
   Name the output column **city**. Order the output ascending by city.
   (You can assume all cities to be the collection of all origin_city or all dest_city)
   (Note: Do not worry if this query takes a while to execute. We are mostly concerned with the results)
   [Output relation cardinality: 3 or 4, depending on what you consider to be the set of all cities]

6. List the names of carriers that operate flights from Seattle to San Francisco, CA. Return each carrier's name only once. Use a nested query to answer this question. (7 points)
   Name the output column **carrier**. Order the output ascending by carrier.
   [Output relation cardinality: 4]

7. Express the same query as above, but do so without using a nested query. Again, name the output column **carrier** and order ascending by carrier. (8 points)
   [Output relation cardinality: 4]

D. Using a Cloud Service (10 points)

The DBMS that we use in this assignment is running somewhere in one of Microsoft's data centers. Comment on your experience using this DBMS cloud service. What do you think about the idea of offering a DBMS as a service in a public cloud?

Save your answer in a file called hw3-d.txt in the submission directory.

Submission Instructions

Please make sure that

- You are submitting the script files directly to Gradescope
- Your file names match the expected file names (hw3-q1.sql, hw3-q2.sql, …, hw3-q7.sql, hw3-d.txt)