

Azure Service

Azure Scale Analytics Workshop

Day 1

Cosmos DB Real-Time Advanced Analytics

이 문서는 데이터솔루션과 마이크로소프트가 공동으로 제작한 자료입니다.

실습 환경 설정

○ 요구 사항

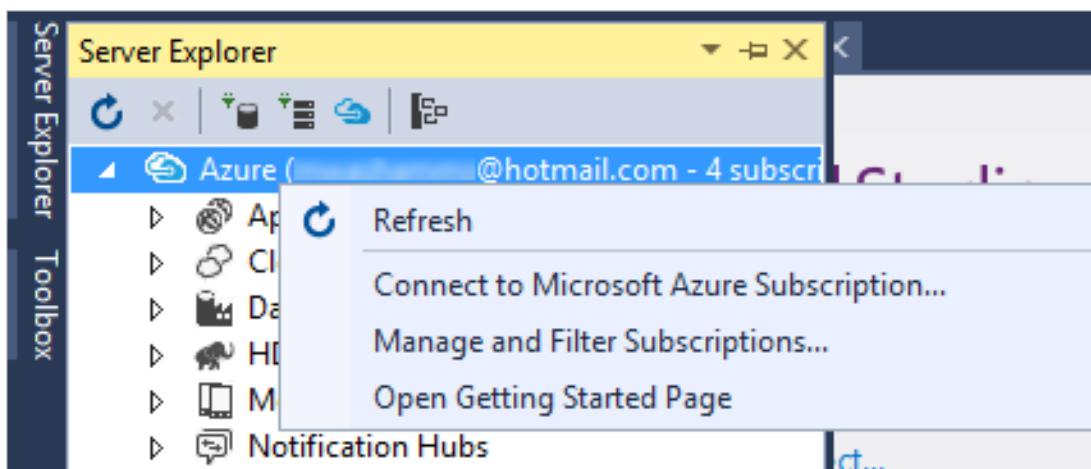
1. Microsoft Azure 구독 (Microsoft 이외의 구독은 유료 구독이어야 합니다).
2. **중요** :이 실습의 OAuth 2.0 액세스 구성 요소를 완료하려면 Azure 구독 내에 Azure Active Directory 내에 앱 등록 및 서비스 주체를 만들 수 있는 권한이 있어야 합니다.

○ 실습 전에

[소요 시간 : 20 분]

작업 1 : Azure에 대한 연결 확인

1. 가상 머신 내에서 Visual Studio 를 시작하고 메시지가 표시되면 Microsoft 계정으로 로그인 할 수 있는지 확인하십시오.
2. Azure 구독에 대한 연결을 확인하려면 **보기** 메뉴에서 **서버 탐색기**를 열고 Azure 구독에 연결할 수 있는지 확인하십시오.

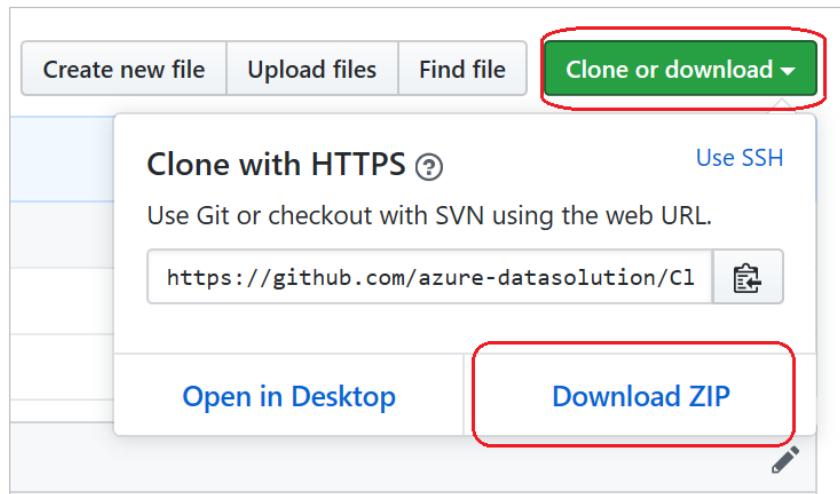


작업 2 : 스타터 파일 다운로드

랩 솔루션에서 처리하기 위해 실시간 지불 데이터와 랩에서 사용되는 데이터 파일을 전송하는 지불 데이터 생성기가 포함 된 스타터 프로젝트를 다운로드 하십시오.

1. LabVM 에서 GitHub 저장소의 .zip 사본을 다운로드하여 스타터 파일을 다운로드하십시오.
2. 웹 브라우저에서 [Cloud Scale Analytics Hands on Lab repo](#) 로 이동하십시오.

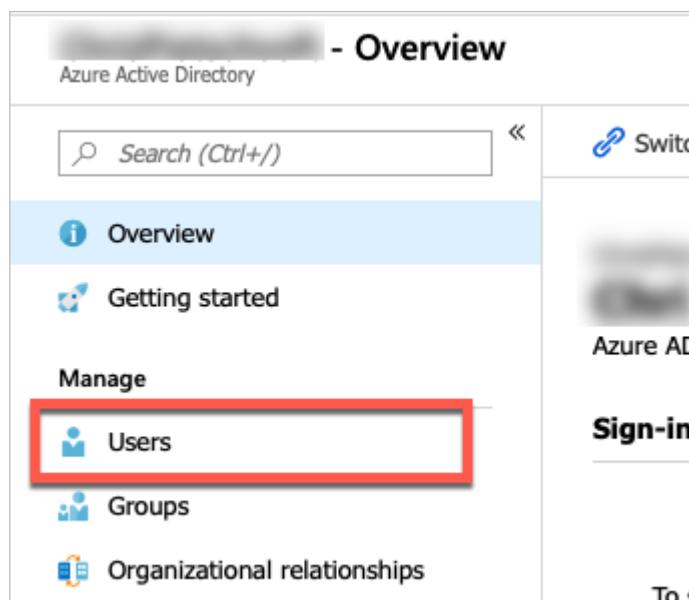
3. 저장소 페이지에서 **복제** 또는 **다운로드** 를 선택한 다음 **ZIP 다운로드**를 선택하십시오.



4. 폴더의 내용을 다음 경로에 압축 해제하십시오 **C:\CloudScaleAnalytics**.

작업 3 : 환경 배포

1. Azure Portal에서 **Azure Active Directory**로 이동한 다음 관리에서 **사용자**를 선택합니다.



2. 현재 Azure Portal에 로그인한 사용자를 선택하십시오.

The screenshot shows the 'Users - All users' page in the Azure Active Directory. On the left, there's a sidebar with navigation links: 'All users' (selected), 'Deleted users', 'Password reset', 'User settings', 'Activity' (Sign-ins, Audit logs), 'Troubleshooting + Support', and 'Troubleshoot'. The main area has a search bar for 'Name' and a 'Show' dropdown set to 'All users'. A table lists users with columns for 'NAME' and 'USER NAME'. The user 'chris' is highlighted with a red box. Other users listed include 'CH' (with a pink profile picture) and 'CP' (with a purple profile picture).

3. 개체 ID를(Object ID)를 복사해 놓습니다.

- Profile

User

Manage

- Profile
- Directory role
- Groups
- Applications
- Licenses
- Devices
- Azure resources
- Authentication methods

Activity

- Sign-ins
- Audit logs

Troubleshooting + Support

- Troubleshoot
- New support request

Identity edit

Name	First name
User name	User type
Object ID	Source

b3...la...

Job info edit

4. 리소스를 생성에서 템플릿 배포(Template Deployment)을 검색창에 입력하세요.

Microsoft Azure

Create a resource 1

Home 2

Dashboard

All services

Dashboard > New

New

template deployment

Azure Marketplace See all

5. 사용자정의 배포(Custom Deployment) 블레이드에서, '편집기에서 자신의 템플릿을 빌드(Build your own template in the editor)' 링크를 선택합니다.

Custom deployment
Deploy from a custom template

Learn about template deployment

[Read the docs](#)

[Build your own template in the editor](#)

Common templates

[Create a Linux virtual machine](#)

6. 템플릿 편집 블레이드에서, ARM 템플릿을 선택하기 위해 **Load File** 버튼을 누릅니다.

Edit template
Edit your Azure Resource Manager template

[Add resource](#) [Quickstart template](#) [Load file](#) [Download](#)

[Parameters \(0\)](#)

```
1 {  
2   "$schema": "htt
```

7. C:\CloudScaleAnalytics\Hands on Lab\Day2\Deployment\environment-template.json
ARM 템플릿을 선택하십시오.
8. 저장을 클릭하십시오.

Edit template

Edit your Azure Resource Manager template

Add resource Quickstart template Load file Download

Parameters (1)
Variables (17)
Resources (10)

[variables('cosmosdb_database...')]
[concat(variables('cosmosdb_d...')]
[concat(variables('cosmosdb_d...')]
[variables('datalake_storageAc...')]
[variables('eventhub_namespa...')]
[concat(variables('eventhub_na...')]
[concat(variables('eventhub_na...')]
[concat(variables('eventhub_na...')]
[variables('databricks_name')] ...
[variables('keyvault_name')] (...)

```
1 {
2     "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
3     "contentVersion": "1.0.0.0",
4     "parameters": {
5         "KeyVaultAccessPolicyUserId": {
6             "type": "string",
7             "defaultValue": "",
8             "metadata": {
9                 "description": "ObjectId for the user who has access to the Key Vault"
10            }
11        }
12    },
13    "variables": {
14        "name_suffix": "[uniqueString(resourceGroup(), 'test')]",
15        "primary_region": "East US",
16        "secondary_region": "West US",
17
18        "cosmosdb_databaseAccount_name": "[variables('cosmosdb_databaseAccount_name')]",
19        "cosmosdb_region": "[variables('cosmosdb_region')]",
20        "cosmosdb_region_secondary": "[variables('cosmosdb_region_secondary')]",
21        "cosmosdb_databaseName": "WoodgroveTest",
22        "cosmosdb_containerName": "transactions"
23    }
24}
```

Save Discard

9. 다음 값을 입력하십시오.

- 자원 그룹 : 랩에 대해 이전에 작성된 자원 그룹을 선택하십시오.
 - 키 저장소 액세스 정책 사용자 개체 ID : 이전에 Azure Active Directory에서 복사한 사용자 개체 ID에 붙여 넣기

Custom deployment
Deploy from a custom template

TEMPLATE

Customized template
10 resources

Edit template Edit param... Learn more

BASICS

* Subscription: [dropdown]

* Resource group: **hands-on-lab** [dropdown] (highlighted with red box)

* Location: (US) West US [dropdown]

SETTINGS

Key Vault Access Policy User Object Id: b3 d7 [dropdown] (highlighted with red box)

TERMS AND CONDITIONS

10. 체크 동의를 ... 확인란을 클릭 한 다음 구매를 .

TERMS AND CONDITIONS

[Azure Marketplace Terms](#) | [Azure Marketplace](#)

By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

I agree to the terms and conditions stated above

Purchase

11. 배포에는 5-10 분이 소요될 수 있습니다.

Microsoft.Template - Overview

Deployment

Search (Ctrl+ /) Delete Cancel Redeploy Refresh

Overview

Inputs Outputs Template

... Your deployment is underway

Check the status of your deployment, manage resources, or troubleshoot deployment issues. Pin this page to your dashboard to easily find it next time.

Deployment name: Microsoft.Template
Subscription: [REDACTED]
Resource group: hands-on-lab

DEPLOYMENT DETAILS (Download)
Start time: 5/12/2019, 12:57:19 PM
Duration: 41 seconds
Correlation ID: [REDACTED]

RESOURCE	TYPE	STATUS	OPERATION DETAILS
woodgrove-db-wsz7okrr...	Microsoft.DocumentDB/...	OK	Operation details
woodgrove-wsz7okrr7u7sc	Microsoft.KeyVault/vaults	OK	Operation details
cosmosdb-mcw-wsz7okr...	Microsoft.Databricks/wo...	Created	Operation details
woodgrove-wsz7okrr7u...	Microsoft.EventHub/na...	OK	Operation details
adlsgen2stgwsz7okrr7u...	Microsoft.Storage/stora...	OK	Operation details

실습을 수행하기 전에 제공된 모든 단계를 따라야 합니다 .

실습 랩 가이드

Day 2 코스모스 DB 실시간 고급 분석

실습 랩 단계별

내용

Day 2 Cosmos DB 실시간 고급 분석 실습 실습 단계별

학습 목표 및 요약

- 개요
- 솔루션 아키텍처
- 요구 사항
- 연습 1 : 스트리밍 트랜잭션 데이터 수집
 - 작업 1 : Cosmos DB로 스트리밍 데이터 수집
- 연습 2 : 대규모 거래 데이터 이해 및 준비
 - 작업 1 : ADLS Gen2 파일 시스템에 대한 OAuth 액세스를 위한 서비스 주체 만들기
 - 작업 2 : Azure Key Vault에 서비스 사용자 자격 증명 및 테넌트 ID 추가
 - 작업 3 : 키 저장소에서 ADLS Gen2 저장소 계정 구성
 - 작업 4 : Key Vault에서 Cosmos DB 키 구성
 - 작업 5 : Azure Databricks 클러스터 만들기
 - 작업 6 : Azure Databricks 열기 및 랩 탑 노트북
 - 작업 7 : Azure Databricks 주요 자격 증명 지원 비밀 구성
 - 작업 8 : Databricks에 Azure Cosmos DB Spark 커넥터 및 scikit-learn 라이브러리 설치
 - 작업 9 : Azure Databricks 및 Spark를 사용하여 기록 트랜잭션 데이터 탐색
 - 작업 10 : Azure Databricks에서 Cosmos DB 변경 피드 및 Spark 구조적 스트리밍을 사용하여 스트리밍 트랜잭션에 응답
- 연습 3 : 사기 모델 작성 및 평가
 - 작업 1 : Databricks에 AzureML 및 Scikit-Learn 라이브러리 설치
 - 작업 2 : 스코어링 웹 서비스 준비 및 배포
 - 작업 3 : 배치 스코어링 모델 준비
- 연습 4 : 배치 점수 데이터 배포
 - 작업 1 : Cosmos DB를 사용하여 배치 점수 데이터 배포

- 작업 2 : Azure Databricks 작업을 사용하여 일정에 따라 트랜잭션 점수를 일괄 처리
- 연습 5 :보고
 - 작업 1 : 사기 추세를 요약하고 시각화하기 위해 Power BI 활용
 - 작업 2 : Azure Databricks에서 대시 보드 만들기
- 연습 후
 - 작업 1 : 리소스 그룹 삭제

Cosmos DB 실시간 고급 분석 연습 단계별 학습목표 및 요약

상거래를 위한 지불 처리 서비스를 제공하는 Woodgrove 은행은 혁신적인 사기 탐지 솔루션의 PoC 를 설계하고 구현하려고 합니다. 이들은 가맹점 고객에게 새로운 서비스를 제공하여 기계 학습 및 고급 분석을 적용하여 사기 거래를 감지함으로써 비용을 절감 할 수 있도록 돕고자합니다. 고객은 전 세계에 있으며 올바른 솔루션은 가능한 한 많은 솔루션을 고객이 서비스를 사용하는 지역에 최대한 가깝게 배포함으로써 서비스 사용 경험이 지연되는 것을 최소화합니다.

이 실습 랩 세션에서는 Woodgrove 은행의 요구를 지원할 수 있는 데이터 파이프 라인의 PoC 를 구현합니다.

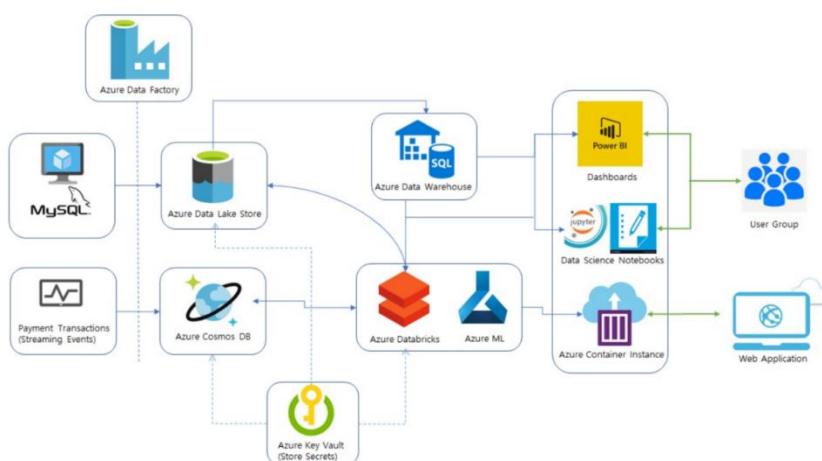
이 워크샵이 끝나면 확장 가능한 머신 러닝, 빅 데이터 및 데이터 처리와 함께 높은 처리량 수집, 낮은 대기 시간 제공 및 글로벌 규모가 필요한 고급 분석 솔루션을 지원하여 Cosmos DB 의 장점을 활용하는 솔루션을 보다 효과적으로 구현할 수 있습니다. 실시간 처리 기능.

○ 개요

상거래를 위한 지불 처리 서비스를 제공하는 Woodgrove 은행은 혁신적인 사기 탐지 솔루션의 개념 증명 (PoC)을 설계하고 구현하려고 합니다. 이들은 가맹점 고객에게 새로운 서비스를 제공하여 기계 학습 및 고급 분석을 적용하여 사기 거래를 감지함으로써 비용을 절감 할 수 있도록 돕고자 합니다. 고객은 전 세계에 있으며 올바른 솔루션은 가능한 한 많은 솔루션을 고객이 서비스를 사용하는 지역에 최대한 가깝게 배포함으로써 서비스 사용 경험이 지연되는 것을 최소화합니다.

○ 솔루션 아키텍처

아래는 이 실습에서 구축할 솔루션 아키텍처의 다이어그램입니다.



솔루션은 Azure Cosmos DB 에 트랜잭션을 쓰는 결제 트랜잭션 시스템으로 시작합니다. Cosmos DB 의 기본 제공 변경 피드 기능을 사용하여 트랜잭션을 사용하여 Azure Databricks 노트북 내에서 들어오는 데이터 스트림으로 읽을 수 있습니다. azure-cosmosdb-spark 커넥터에 연결하고 Azure Data Lake Storage 가 지원하는 Azure Databricks Delta 테이블 내에 장기적으로 저장됩니다. 델타 테이블은 트랜잭션 데이터에 대한 삽입 및 업데이트 (예 : 업 서트)를 효율적으로 관리합니다. 비즈니스 분석가는 Power BI 의 Spark 커넥터를 사용하여 Power BI 의 대시 보드 및 보고서를 사용하여 이 데이터에 대해 Databricks 에서 작성된 테이블에 액세스 할 수 있습니다. 또는 의미 분석 모델을 Azure Analysis Service 에 저장하여 데이터를 Power BI 에 제공 할 수 있으므로 보고를 위해 전용 Databricks 클러스터를 계속 실행할 필요가 없습니다. 데이터 과학자와 엔지니어는 Azure Databricks 노트북을 사용하여 Databricks 테이블에 대한 자체 보고서를 만들 수 있습니다.

Azure Databricks 는 Azure Data Lake Storage 에 저장된 기록 데이터를 사용하여 기계 학습 모델의 교육 및 유효성 검사를 지원합니다. 델타 테이블 또는 다른 히스토리 테이블에 저장된 데이터를 사용하여 모델을 주기적으로 재 훈련 할 수 있습니다. Azure Machine Learning 서비스는 학습 모델을 고 가용성 Azure Kubernetes Service 클러스터 (AKS 클러스터)에서 실행되는 실시간 스코어링 웹 서비스로 배포하는데 사용됩니다. 훈련 된 모델은 또한 Databricks 작업을 통한 예약된 오프라인 스코어링에 사용되며 "의심스러운 활동" 출력은 Azure Cosmos DB 에 저장되므로 웹 응용 프로그램을 통해 Woodgrove 은행 고객과 가장 가까운 지역에서 전 세계적으로 사용할 수 있습니다.

마지막으로 Azure Key Vault 는 계정 키 및 연결 문자열과 같은 비밀을 안전하게 저장하는데 사용되며 Azure Databricks 비밀 범위의 백업 역할을 합니다.

참고 : 기본 솔루션은 가능한 많은 실행 가능한 접근 방법 중 하나 일뿐입니다.

○ 요구 사항

1. Microsoft Azure 구독 (Microsoft 이외의 구독은 유료 구독이어야 합니다.)
2. Databricks Runtime 5.1 이상을 실행하는 Azure Databricks 클러스터 Azure Data Lake Storage Gen2 와의 Azure Databricks 통합 은 **Databricks Runtime 5.1** 에서 완전히 지원됩니다 .
 - 중요 :이 실습의 OAuth 2.0 액세스 구성 요소를 완료하려면 다음을 수행해야 합니다.
 - Databricks Runtime 5.1 이상을 실행하는 클러스터가 있어야 합니다.
 - Azure 구독 내에서 Azure Active Directory 내에 앱 등록 및 서비스 주체를 만들 수 있는 권한이 있습니다

연습 1 : 스트리밍 트랜잭션 데이터 수집

[소요 시간 : 30 분]

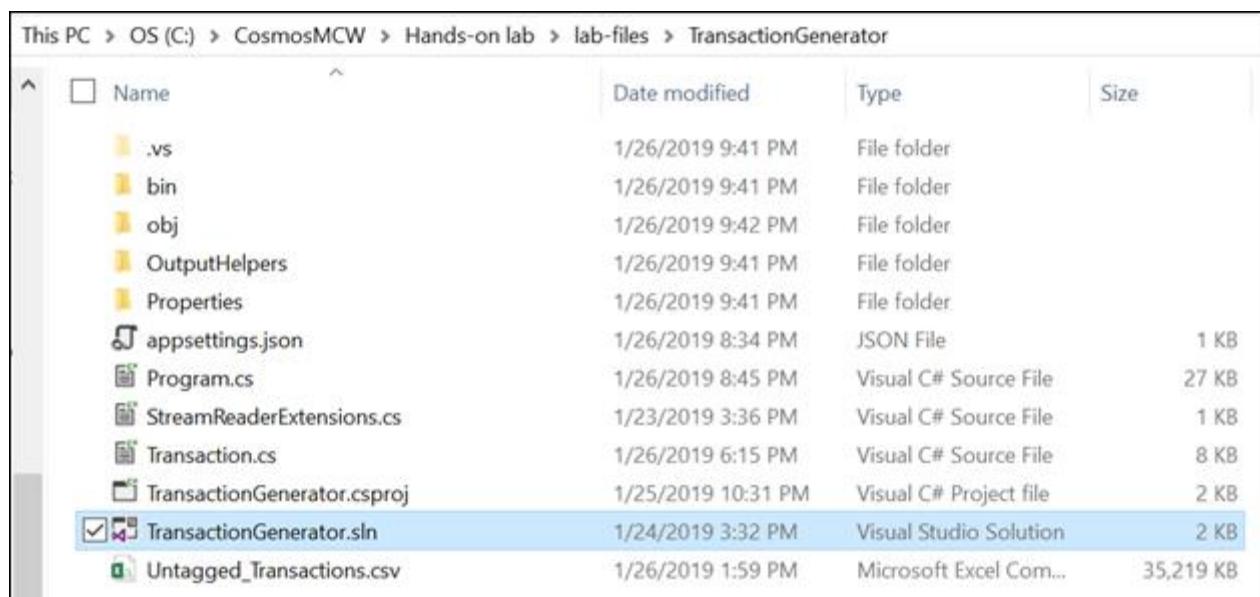
이 연습에서는 실시간 스트리밍 온라인 결제를 Azure Cosmos DB에 쓰도록 결제 트랜잭션 생성기를 구성합니다. 결국 생성된 데이터를 처리할 다음 실습을 계속하기 전에 최상의 수집 옵션을 선택하게 됩니다.

태스크 1 : 트랜잭션 생성기 구성

이 태스크에서는 소스 코드에서 연결 정보를 추가하여 지불 트랜잭션 데이터 생성기 프로젝트를 구성합니다.

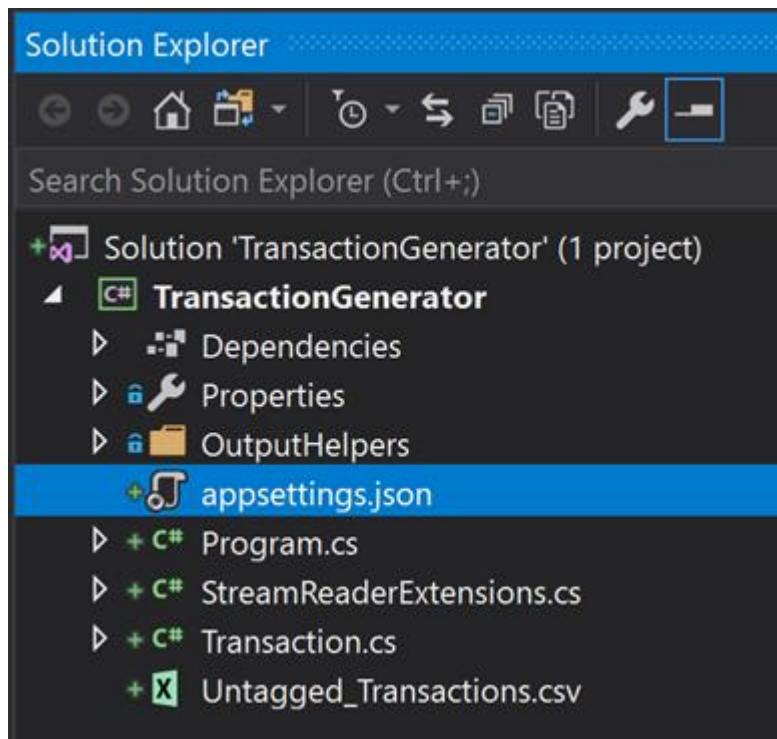
- VM에서 파일 탐색기를 열고 CloudScaleAnalytics-master.zip 파일을 추출한 위치 ([C:\CloudScaleAnalytics](#))로 이동합니다.
- [Hands on Lab\Day2\lab-files\TransactionGenerator](#)

디렉토리의 **TransactionGenerator.sln** 을 더블클릭하면 Visual Studio에서 솔루션이 열립니다.



Name	Date modified	Type	Size
.vs	1/26/2019 9:41 PM	File folder	
bin	1/26/2019 9:41 PM	File folder	
obj	1/26/2019 9:42 PM	File folder	
OutputHelpers	1/26/2019 9:41 PM	File folder	
Properties	1/26/2019 9:41 PM	File folder	
appsettings.json	1/26/2019 8:34 PM	JSON File	1 KB
Program.cs	1/26/2019 8:45 PM	Visual C# Source File	27 KB
StreamReaderExtensions.cs	1/23/2019 3:36 PM	Visual C# Source File	1 KB
Transaction.cs	1/26/2019 6:15 PM	Visual C# Source File	8 KB
TransactionGenerator.csproj	1/25/2019 10:31 PM	Visual C# Project file	2 KB
TransactionGenerator.sln	1/24/2019 3:32 PM	Visual Studio Solution	2 KB
Untagged_Transactions.csv	1/26/2019 1:59 PM	Microsoft Excel Com...	35,219 KB

- [appsettings.json](#) 솔루션 탐색기를 두 번 클릭하여 엽니다. 이 파일에는 콘솔 앱에서 Azure 서비스에 연결하고 응용 프로그램 동작 설정을 구성하는 데 사용되는 설정이 포함되어 있습니다. 콘솔 앱은 이 파일에 저장된 값을 사용하거나 시스템의 환경 변수 내에 사용되도록 프로그래밍되어 있습니다. 이를 통해 실행 파일을 배포하거나 컨테이너화하고 명령 줄을 통해 환경 변수를 전달할 수 있습니다.



appsettings.json 파일은 다음 내용이 포함되어 있습니다. :

```
{  
    "COSMOS_DB_ENDPOINT": "",  
    "COSMOS_DB_AUTH_KEY": "",  
  
    "SECONDS_TO_LEAD": "0",  
    "SECONDS_TO_RUN": "600",  
    "ONLY_WRITE_TO_COSMOS_DB": "true"  
}
```

SECONDS_TO_LEAD 결제 거래 데이터를 보내기 전에 대기하는 시간입니다. 기본값은 0 입니다.

SECONDS_TO_RUN 데이터 전송을 중지하기 전에 생성기가 실행될 수 있는 최대 시간입니다. 기본값은 600 입니다. 포함 된 Untagged_Transactions.csv 파일의 데이터가 전송 된 후에도 데이터 전송이 중지됩니다 .

경우 `ONLY_WRITE_TO_COSMOS_DB` 속성에 설정되어 `true`로 설정합니다. 기본값은 `false`입니다.

파일을 저장하십시오.

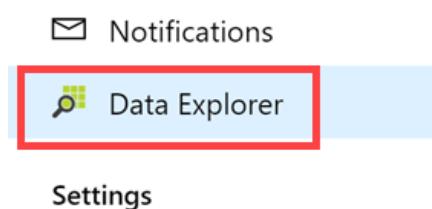
4. Program.cs Visual Studio 솔루션 탐색기에서 엽니다
5. Visual Studio 의 드롭 다운 메뉴에서 보기 를 선택한 다음 작업 목록 을 선택하십시오 .

작업 1 : Cosmos DB 로 스트리밍 데이터 수집

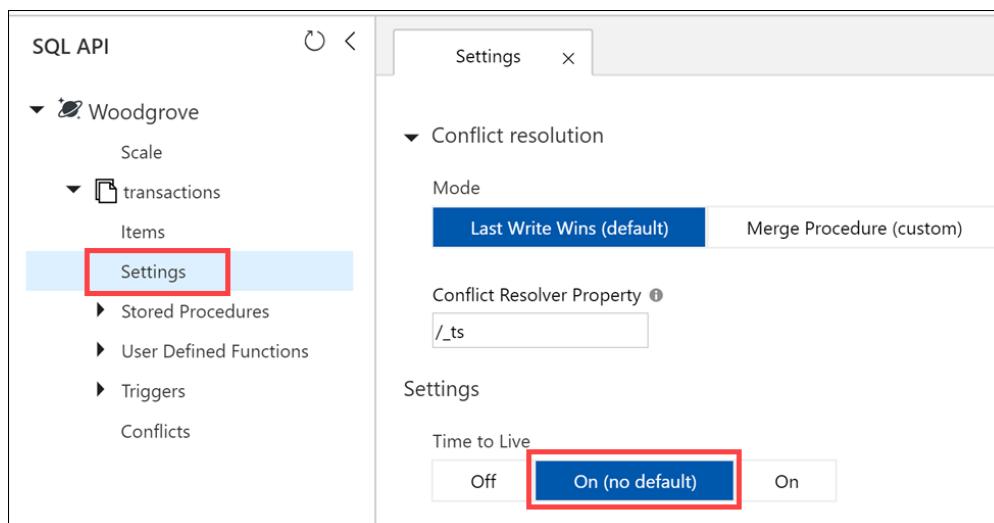
이 작업에서는 Cosmos DB 의 TTL (Time-To-Live) 설정을 기본값없이 On 으로 구성합니다. 이를 통해 데이터 생성기 ttl 는 개별 메시지가 전송 될 때 TTL 값 (개체 속성)을 설정하여 원하는 시간이 지나면 수집 된 메시지를 만료하거나 삭제할 수 있습니다.

다음으로 Azure Cosmos DB URI 및 키 값을 데이터 생성기로 전달하여 컬렉션에 연결하고 컬렉션에 이벤트를 보낼 수 있습니다.

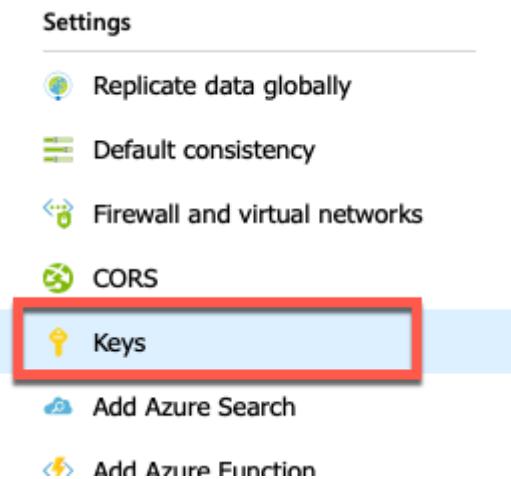
1. Azure Portal 에서 Azure Cosmos DB 계정으로 이동한 다음 왼쪽 메뉴에서 데이터 탐색기를 선택 합니다.



2. Woodgrove 데이터베이스와 트랜잭션 컨테이너를 확장 한 다음 설정을 선택하십시오.
3. 설정 블레이드 내의 설정에서 TTL (Time to Live)에 대해 켜기 (기본값 없음) 옵션을 선택합니다. 컨테이너에 추가 된 문서를 자체 TTL 값으로 구성하려면 이 설정이 필요합니다.



4. 저장 을 선택 하여 설정을 적용하십시오.
5. 온 애저 코스모스 DB 계정 블레이드, 선택 키 에서 설정 .



6. Cosmos DB 의 엔드 포인트 URI 및 기본 키를 복사하십시오 . 나중에 사용하기 위해 이 값을 메모장 또는 이와 유사한 곳에 저장하십시오.

The screenshot shows the 'Keys' blade for an Azure Cosmos DB account. It displays the primary key and connection strings. The primary key is highlighted with a red box. The connection strings show the endpoint and account key.

Primary Key	Value
EOZrLRl8O8tdpivArOcoSlvyc0wqccpyMV3ubhXQHfebyy0InCfaCJ2HRmy8tthPS00NpF6eihyi7n215nh03A==	

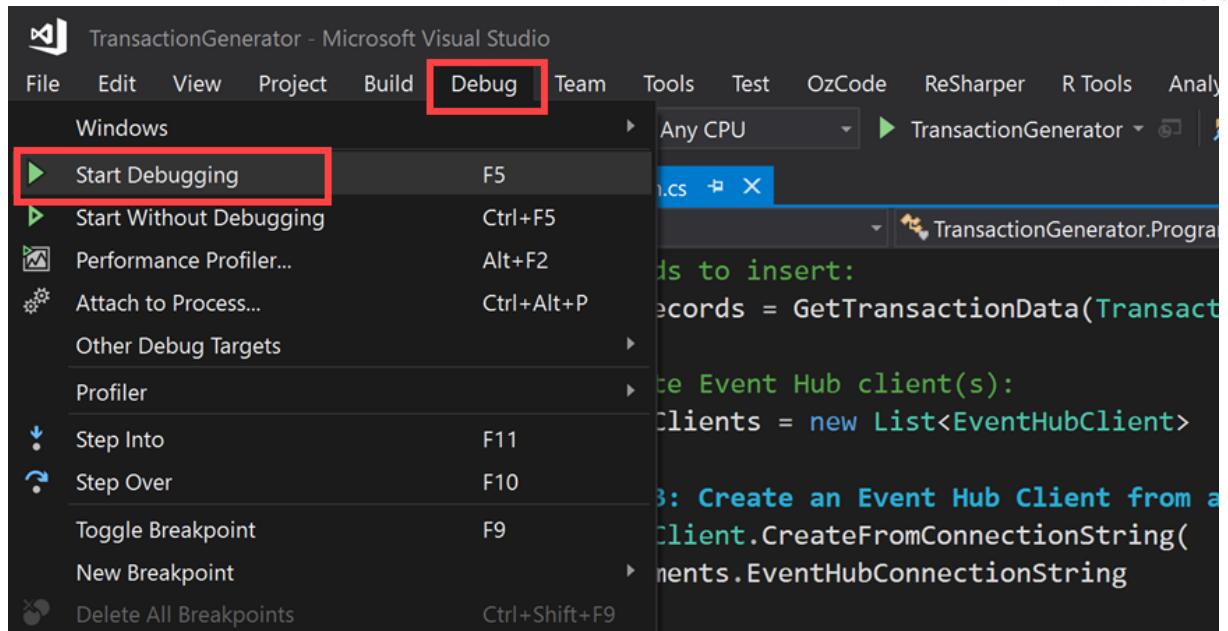
Secondary Connection String	Value
AccountEndpoint=https://woodgrove-db-wsz7okrr7u7sq.documents.azure.com:443/;AccountKey=EOZrLRl8O8tdpivArOcoSlvyc0wqccpyMV3ubhXQHfebyy0InCfaCJ2HRmy8tthPS00NpF6eihyi7n215nh03A==	

7. Visual Studio 를 열어 TransactionGenerator 프로젝트로 돌아갑니다.
8. appsettings.json 을 열은 후 COSMOS_DB_ENDPOINT 와 COSMOS_DB_AUTH_KEY 를 넣습니다. 예를 들면 다음과 같습니다.

```
"COSMOS_DB_ENDPOINT": "https://woodgrove-db.documents.azure.com:443/",  
"COSMOS_DB_AUTH_KEY": "X40BHKQtbn0kryyA3LayCqe8TmGok63SzS07eYtn5UCihxPWVY801FdM6"
```

9. 변경 사항을 저장하십시오.

10. Visual Studio 의 최상위 메뉴에서 디버그 를 클릭 한 다음 디버깅 시작 을 클릭하여 콘솔 앱을 실행하거나 키보드에서 F-5 를 누르십시오.



11. PaymentGenerator 콘솔 창이 열리고 몇 초 후에 데이터 전송이 시작되는 것을 볼 수 있습니다. 당신은 창을 누르거나 닫을 수 있습니다 **Ctrl+C** 또는 **Ctrl+Break** 를 이용해 코스모스 DB 에 데이터 전송을 언제든지 중지할 수 있습니다.

```

Payment Generator
=====
Press Ctrl+C or Ctrl+Break to cancel.
Statistics for generated payment data will be updated for every 1000 sent

Retrieving sample transaction data...
Sample transaction data retrieved. 198999 records found.
Found collection 'transactions' with 15000 RU/s (15000 reads/second; 3000 writes/second @ 1KB doc size)
The collection will cost an estimated $1.25 per hour ($900.00 per month (per write region))
Total requests: requested 1000

Event Hub: inserted 993 docs @ 92.51 writes/s
Event Hub: processing time 10734 ms (slower)
Event Hub: total elapsed time 10.73 seconds (slower)
Event Hub: total succeeded 993
Event Hub: total pending 0
Event Hub: total failed 0

Cosmos DB: inserted 995 docs @ 166.83 writes/s, 7463.79 RU/s (19.35B max monthly 1KB writes)
Cosmos DB: processing time 5964 ms (faster)
Cosmos DB: total elapsed time 5.96 seconds (faster)
Cosmos DB: total succeeded 995
Cosmos DB: total pending 0
Cosmos DB: total failed 0

Total requests: requested 2000

Event Hub: inserted 1002 docs @ 211.13 writes/s
Event Hub: processing time 4746 ms (faster)
Event Hub: total elapsed time 15.48 seconds (slower)
Event Hub: total succeeded 1995
Event Hub: total pending 0
Event Hub: total failed 0

Cosmos DB: inserted 1000 docs @ 201.61 writes/s, 9075.14 RU/s (23.52B max monthly 1KB writes)
Cosmos DB: processing time 4960 ms (slower)
Cosmos DB: total elapsed time 10.92 seconds (faster)
Cosmos DB: total succeeded 1995
Cosmos DB: total pending 0
Cosmos DB: total failed 0

Total requests: requested 3000

Event Hub: inserted 1001 docs @ 239.42 writes/s
Event Hub: processing time 4181 ms (faster)
Event Hub: total elapsed time 19.66 seconds (slower)
Event Hub: total succeeded 2996
Event Hub: total pending 0
Event Hub: total failed 0

Cosmos DB: inserted 1000 docs @ 217.91 writes/s, 9838.46 RU/s (25.50B max monthly 1KB writes)
Cosmos DB: processing time 4589 ms (slower)
Cosmos DB: total elapsed time 15.51 seconds (faster)
Cosmos DB: total succeeded 2995
Cosmos DB: total pending 0

```

출력 맨 위에는 작성된 Cosmos DB 컨테이너 (트랜잭션), 요청 된 RU / s 및 예상 시간별 및 월별 비용에 대한 정보가 표시됩니다. 1,000 개의 레코드를 보내도록 요청한 후 출력 통계가 표시됩니다.

- 삽입 된 줄은 이 배치의 성공적인 삽입 및 RU / s 사용량 및 Cosmos DB 통계에 추가 된 예상 월간 수집 속도로 초당 쓰기 처리량을 보여줍니다.
- 처리 시간 : 지난 1,000 개의 요청 된 삽입에 대한 처리 시간이 다른 서비스보다 빠르거나 느린지 여부를 표시합니다.
- 총 경과 시간 : 모든 문서를 처리하는 데 걸린 총 시간입니다.
- Cosmos DB에서 이 값이 계속 높아지면 Cosmos DB 요청이 조절되고 있음을 나타내는 좋은 지표입니다. 컨테이너의 RU 를 늘리십시오.
- 성공은 서비스에 누적 된 성공적인 삽입 수를 표시합니다.
- 보류는 벌크 헤드 대기열에 있는 항목입니다. 서비스가 수요를 충족시킬 수 없는 경우이 금액은 계속 증가합니다.
- 예외가 발생한 누적 실패한 요청입니다.

12. 실험적으로 Cosmos DB 컨테이너에 대해 요청 된 RU / s 수를 700 으로 줄이십시오. 이렇게하면 조절로 인해 Cosmos DB 로의 전송 속도가 점점 느려질 것입니다. 또한 보류중인 큐가 더 높은 속도로 증가하는 것을 볼 수 있습니다. 그 이유는 쓰기 횟수 (쓰기가 일반적으로 5 RU / s 를 사용하고 1KB 크기의 문서를 읽는 데 1RU / s 만 사용함)가 할당 된 RU / s 를 초과하면 Cosmos DB 가 소비자에게 리소스가 제한되어 있음을 알리기 위해 *retry_after* 헤더 값이 있는 429 응답 . SDK 는 지정된 시간 동안 기다렸다가 다시 시도하여 이를 자동으로 처리합니다. 실험이 끝나면 RU 를 15,000 으로 다시 설정하십시오.

연습 2 : 대규모 거래 데이터 이해 및 준비

[소요 시간 : 45 분]

이 연습에서는 Databricks 작업 영역에서 ADLS Gen2 및 Cosmos DB 로의 연결을 작성 합니다. 그런 다음 Azure Databricks 를 사용하면 Woodgrove 에서 제공 한 일부 기존 원시 트랜잭션 데이터를 가져 와서 탐색하여 데이터를 사용하여 기계 학습 모델을 구축 및 교육하기 전에 수행해야하는 준비 사항을 더 잘 이해할 수 있습니다. 그런 다음 Databricks 에서 Cosmos DB 에 대한 연결을 사용하여 Cosmos DB 변경 피드에서 직접 스트리밍 트랜잭션을 읽습니다. 마지막으로 들어오는 스트리밍 트랜잭션 데이터를 데이터 레이크에 저장된 Azure Databricks Delta 테이블에 씁니다.

작업 1 : ADLS Gen2 파일 시스템에 대한 OAuth 액세스를 위한 서비스 주체 만들기

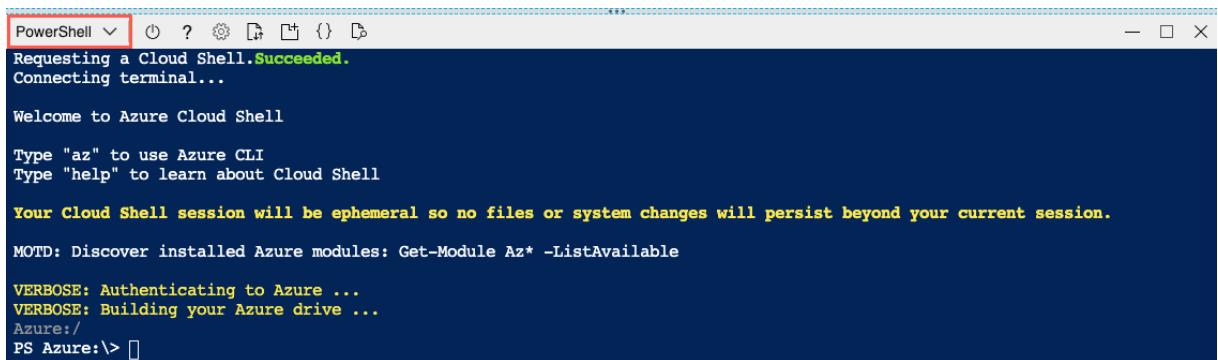
Databricks 를 사용하여 ADLS Gen2 파일 시스템에 액세스 할 때 추가 보안 계층으로 OAuth 2.0 을 사용하여 인증 할 수 있습니다. 이 작업에서는 Azure CLI 를 사용하여 서비스 주체라고하는 Azure AD (Azure Active Directory)에서 ID 를 만들어 OAuth 인증을 쉽게 사용할 수 있습니다.

중요 :이 작업을 완료하려면 Azure 구독 내에 App 등록 및 서비스 주체를 만들려면 Azure 구독 내 권한이 있어야합니다.

1. [애저 포털](#) 에서, 선택 클라우드 웰 상단 도구 모음에서 아이콘을.



2. Cloud Shell 창에서 **PowerShell** 이 선택되어 있는지 확인하십시오 .



3. 다음으로 **woodgrove-sp** 라는 서비스 주체를 만들고 **ADLS Gen2 저장소 계정** 의 *Storage Blob Data Contributor* 역할에 할당 하는 명령을 실행 합니다 . 명령 형식은 다음과 같습니다.

```
az ad sp create-for-rbac -n "woodgrove-sp####" --role "Storage Blob Data Contributor" --scopes {adls-gen2-storage-account-resource-id}
```

중요 : {adls-gen2-storage-account-resource-id}값을 ADLS Gen2 저장소 계정의 리소스 ID 로 바꿔야 합니다. #####는 개인 고유번호로 수정해야 합니다.

4. 위에서 교체해야하는 ADLS Gen2 저장소 계정 리소스 ID 를 검색하려면 Azure 탐색 메뉴에서 **리소스 그룹** 으로 이동하고 필터 상자에 "hands-on-lab-SUFFIX"를 입력 한 다음

Hands-on-lab-SUFFIX 를 선택하십시오. 목록에서 자원 그룹.

- Hands-on-Lab 자원 그룹에서 ADLS 세대 스토리지를 선택, 이전에 제공된 계정 및 ADLS 세대 스토리지 계정 블레이드 선택 **속성**에서 **설정** 왼쪽 메뉴에서 **저장소 계정 리소스 ID** 값의 오른쪽 클립 보드 버튼에 복사를 선택합니다

aldsgen2store - Properties
Storage account

Settings

- Access keys
- Geo-replication
- CORS
- Configuration
- Encryption
- Shared access signature
- Firewalls and virtual networks
- Advanced security
- Properties** (highlighted)
- Locks

Storage account resource ID: /subscriptions/30fc406c-c745-44f0-be2d-63b1c860cde0/resourceGroups/hands-on-lab/providers/Mi... (Copy to clipboard button highlighted)

Blob service resource ID: /subscriptions/30fc406c-c745-44f0-be2d-63b1c860cde0/resourceGroups/hands-on-lab/providers/Mi... (Copy to clipboard button highlighted)

Primary Blob Service Endpoint: https://aldsgen2store.blob.core.windows.net/ (Copy to clipboard button highlighted)

File service resource ID: /subscriptions/30fc406c-c745-44f0-be2d-63b1c860cde0/resourceGroups/hands-on-lab/providers/Mi... (Copy to clipboard button highlighted)

Primary File Service Endpoint: https://aldsgen2store.file.core.windows.net/ (Copy to clipboard button highlighted)

- 스토리지 계정 리소스 ID를 위의 명령에 붙여 넣은 다음 업데이트 된 az ad sp create-for-rbac 명령을 복사하여 Cloud Shell 프롬프트에 붙여 넣고 Enter 을 누릅니다 . 이 명령은 구독 ID 및 자원 그룹 이름을 가진 다음과 유사해야합니다.

```
az ad sp create-for-rbac -n "woodgrove-sp####" --role "Storage Blob Data Contributor" --scope /subscriptions/XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXX/resourceGroups/hands-on-lab/providers/Microsoft.Storage/storageAccounts/aldsgen2store
```

```
PS Azure:> az ad sp create-for-rbac -n "woodgrove-sp" --role contributor --scopes /subscriptions/ /resourceGroups/hands-on-lab/providers/Microsoft.Storage/storageAccounts/aldsgen2store
Changing "woodgrove-sp" to a valid URI of "http://woodgrove-sp", which is the required format used for service principal names
{
  "appId": "68b7968d-d0e6-4dbf-83f9-3da68cba4719",
  "displayName": "woodgrove-sp",
  "name": "http://woodgrove-sp",
  "password": "2c2d601b-5b44-4ae9-8cb4-8ad90e533658",
  "tenant": "9c37bff0-cd20-"
}
Azure:/
PS Azure:>
```

7. 다음 단계에서 필요하므로 명령의 출력을 텍스트 편집기로 복사하십시오. 출력은 다음과 유사해야합니다.

```
{
  "appId": "68b7968d-d0e6-4dbf-83f9-3da68cba4719",
  "displayName": "woodgrove-sp",
  "name": "http://woodgrove-sp",
  "password": "2c2d601b-5b44-4ae9-8cb4-8ad90e533658",
  "tenant": "9c37bff0-cd20-XXXX-XXXX-XXXXXXXXXXXX"
}
```

8. 역할 할당을 확인하려면 ADLS Gen2 저장소 계정 블레이드의 왼쪽 메뉴에서 **IAM (액세스 제어)** 을 선택한 다음 **역할 할당** 탭을 선택하고 *STORAGE BLOB DATA CONTRIBUTOR* 역할에서 **woodgrove-sp** 를 찾습니다 .

NAME	TYPE	ROLE	SCOPE
woodgrove-sp	App	Storage Blob Data Contributor	All scopes

작업 2 : Azure Key Vault에 서비스 사용자 자격 증명 및 Tenant ID 추가

1. Azure Databricks 에서 ADLS Gen2 계정에 대한 액세스를 제공하려면 Azure Key Vault 계정에 저장된 비밀을 사용하여 Databricks 내에서 새로 만든 서비스 주체의 자격 증명을 제공합니다. Azure Portal에서 Azure Key Vault 계정으로 이동한 다음 **액세스 정책** 을 선택하고 **+ 새 추가** 버튼을 선택 합니다.

2. 현재 주체로 포털에 로그인 한 계정을 선택하고 , 및 아래에서 **key permissions, secret permissions, certificate permissions** 모두 선택을 선택한 다음 확인을 클릭하고 **저장** 을 클릭 합니다.

3. 이제 왼쪽 메뉴의 설정에서 **비밀** 을 선택하십시오 . 비밀 블레이드 의 상단 도구 모음 에서 **+ 생성 / 가져 오기** 를 선택 합니다.

The screenshot shows the 'woodgrove-vault - Secrets' blade in the Azure portal. On the left, there's a sidebar with 'Settings' and four tabs: 'Keys' (disabled), 'Secrets' (selected and highlighted with a red box), 'Certificates', and 'Access policies'. At the top right, there's a search bar, a 'Generate/Import' button (also highlighted with a red box), and 'Refresh' and 'Restore' buttons. The main content area has a header 'NAME' and a message 'There are no secrets available.'

4. Secrets(비밀) 블레이드 작성에서 다음을 입력하십시오.

- **업로드 옵션** : 수동을 선택하십시오.
- **이름** : "Woodgrove-SP-Client-ID"를 입력하십시오.
- **값** : 이전 단계에서 복사 한 Azure CLI 출력 의 **appId** 값을 붙여 넣습니다.

Create a secret

Upload options
Manual

* Name !
Woodgrove-SP-Client-ID

* Value
.....

Content type (optional)

Set activation date? !

Set expiration date? !

Enabled? Yes No

Create

5. 생성을 선택 하십시오

위와 같은 방법으로 Secrets 에 아래의 표에 있는 이름(Name)과 값(Value)을 입력하고 생성(create)을 합니다.

업로드옵션	이름	값 (전 단계에서 Azure CLI 출력값 불여넣기)
수동	Woodgrove-SP-Client-Key	Password : (Azure CLI 의 출력값)
수동	Azure-Tenant-ID	Tenant : (Azure CLI 의 출력값)

작업 3 : 키 저장소에서 ADLS Gen2 저장소 계정 구성

이 작업에서는 Key Vault 내에서 ADLS Gen2 스토리지 계정의 키를 구성합니다.

1. Azure Portal에서 ADLS Gen2 저장소 계정으로 이동한 다음 왼쪽 메뉴의 설정에서 액세스 키를 선택합니다. 저장소 계정 이름과 키 값을 복사하여 Key Vault 계정에 Secrets(비밀)에 추가합니다.

adlsgen2storage - Access keys

Storage account

Search (Ctrl+ /)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Settings

- Access keys**
- CORS

Use access keys to authenticate your applications when making requests to this Azure storage account. Store your access keys securely - for example, using Azure Key Vault - and don't share them. We recommend regenerating your access keys regularly. You are provided two access keys so that you can maintain connections using one key while regenerating the other.

When you regenerate your access keys, you must update any Azure resources and applications that access this storage account to use the new keys. This action will not interrupt access to disks from your virtual machines. [Learn more](#)

Storage account name
adlsgen2storage

key1

Key
CL0EQtPz5nib6lcmYGcQtU9QwKrjj4rKeeJGwcl1mM6myw+AR6paA9kYtsB2u0DNgHkulvlu...

- 새 브라우저 탭 또는 창을 열고 Azure Portal에서 Azure Key Vault 계정으로 이동한 다음 왼쪽 메뉴의 설정에서 **비밀**을 선택합니다. 비밀 블레이드의 상단 도구 모음에서 **+ 생성 / 가져 오기**를 선택합니다.

woodgrove-vault - Secrets

Key vault

Search (Ctrl+ /)

+ Generate/Import Refresh Restore

Diagnose and solve problems

Settings

- Secrets**
- Keys
- Certificates
- Access policies

NAME	T
There are no secrets available.	

이번 단계도 작업 2와 같이 Key Vault Secrets에 아래의 표에 있는 이름(Name)과 값(Value)을 입력하고 생성(create)을 합니다.

업로드옵션	이름	값 (전 단계에서 복사한 저장소 계정 (storage account)과 키(Key1) 값을 붙여넣기)
-------	----	----------------------------------------------------------

수동	ADLS-Gen2-Account-Name	Storage account name 값 붙여넣기
수동	ADLS-Gen2-Account-Key	storage account key 값 붙여넣기

이름	형식	상태	만료 날짜
ADLS-Gen2-Account-Key		✓ 사용	
ADLS-Gen2-Account-Name		✓ 사용	
Azure-Tenant-ID		✓ 사용	
Cosmos-DB-Key		✓ 사용	
Cosmos-DB-URI		✓ 사용	
Woodgrove-SP-Client-ID		✓ 사용	
Woodgrove-SP-Client-Key		✓ 사용	

작업 4 : Key Vault에서 Cosmos DB 키 구성

- 새 브라우저 탭 또는 창을 열고 Azure Portal에서 Azure Key Vault 계정으로 이동한 다음 왼쪽 메뉴의 설정에서 **비밀**을 선택합니다. 비밀 블레이드의 상단 도구 모음에서 **+ 생성 / 가져 오기**를 선택합니다.

이번 단계도 작업 3와 같이 Key Vault Secrets에 아래의 표에 있는 이름(Name)과 값(Value)을 입력하고 생성(create)을 합니다.

업로드옵션	이름	값 (앞에서 편집기에 복사한 값 붙여넣기)
수동	Cosmos-DB-URI	이전 단계에서 복사한 Azure Cosmos DB URI 값
수동	Cosmos-DB-Key	이전 단계에서 복사한 Azure Cosmos DB Primary Key 값

▶ [Key Vault Secrets] 전체 입력 정보 확인하기

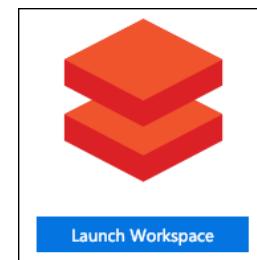
[작업2]부터 [작업4] 단계에서 입력한 Key vault secrets의 전체 정보가 정상적으로 생성이 되었는지 확인을 합니다.



작업 5 : Azure Databricks 클러스터 만들기

이 작업에서는 Azure Databricks 작업 영역에 연결하고이 실습에 사용할 클러스터를 만듭니다.

1. [Azure Portal](#)로 돌아가서 위에서 프로비저닝 한 Azure Databricks 작업 영역으로 이동 하고 개요 블레이드에서 **작업 영역 시작을 선택** 하고 필요한 경우 Azure 자격 증명으로 작업 영역에 로그인합니다.



2. 선택 **클러스터** 왼쪽 탐색 메뉴에서를 누른 다음 선택 + **클러스터를 만듭니다 .**

3. 클러스터 생성 화면에서 다음을 입력하십시오.

- **클러스터 이름** : lab-cluster 와 같은 클러스터 이름을 입력하십시오.
- **클러스터 모드** : 표준을 선택합니다.
- **Databricks 런타임 버전** : 런타임 : 5.5LTS (Scala 2.11, Spark 2.4.3)를 선택하십시오.
- **파이썬 버전** : 선택 3.
- **자동 크기 조정 사용** : 이것이 선택되어 있는지 확인하십시오.
- **XX 분 동안 활동이 없으면 종료** : 이 확인란을 선택된 상태로 두고 분 수를 120 으로 설정하십시오.
- **작업자 유형** : Standard_DS3_v2 를 선택하십시오.
 - **최소 Worker** : 1 로 설정합니다.
 - **최대 Worker** : 2 로 설정하십시오.
- **Driver 유형** : Worker 와 동일하게 설정합니다.
- 고급 옵션을 펼치고 Spark 구성 상자에 다음을 입력하십시오.

```
spark.databricks.delta.preview.enabled true
```

The screenshot shows the 'Create Cluster' page with the following settings:

- New Cluster** button is visible.
- Standard** worker type is selected.
- Databricks Runtime Version**: Runtime: 5.2 (Scala 2.11, Spark 2.4.0).
- Python Version**: 3.
- Autopilot Options**:
 - Enable autoscaling
 - Terminate after 120 minutes of inactivity
- Worker Type**: Standard_DS4_v2 (28.0 GB Memory, 8 Cores, 1.5 DBU), Min Workers: 2, Max Workers: 8.
- Driver Type**: Same as worker (28.0 GB Memory, 8 Cores, 1.5 DBU).
- Advanced Options** section is expanded, showing the configuration parameter:


```
spark.databricks.delta.preview.enabled true
```

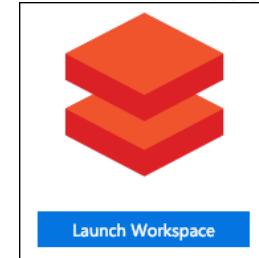
 Below this, tabs for Spark, Tags, Logging, and Init Scripts are visible.

4. 클러스터 작성을 선택 하십시오 . 클러스터를 생성하고 시작하는 데 3-5 분이 걸립니다.

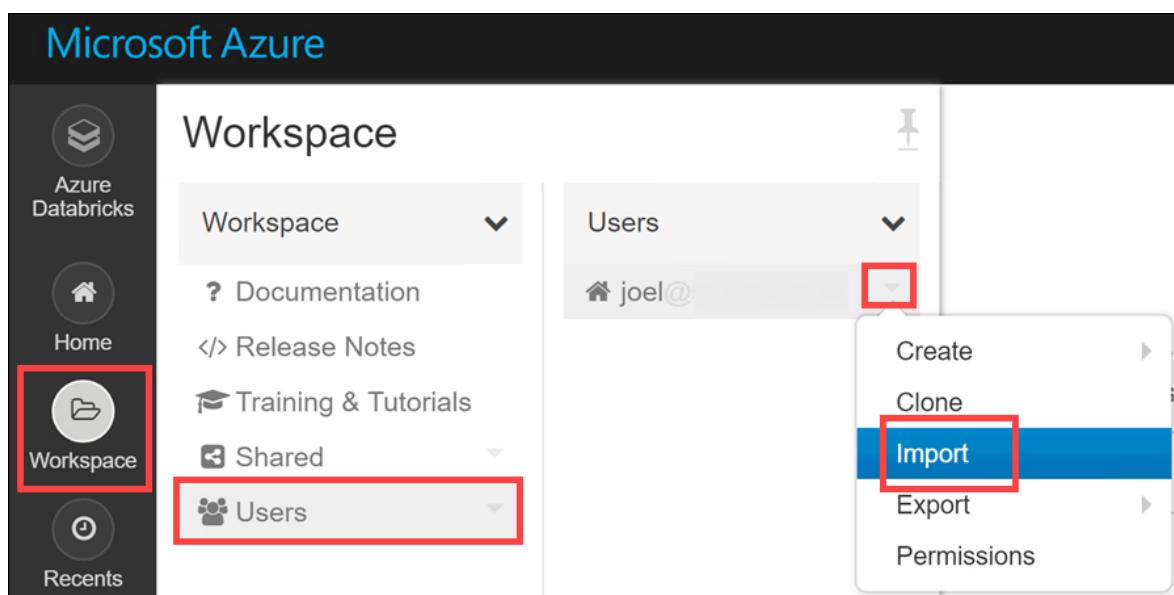
작업 6 : Azure Databricks 열기 및 랩 노트북

이 작업에서는 [Scale Analytics Workshop GitHub](#) 리포지토리에 포함 된 노트북을 Azure Databricks 작업 영역으로 가져옵니다.

- Azure Portal에서 Azure Databricks 작업 영역으로 이동하고
개요 블레이드에서 **작업 영역 시작**을 선택하고 필요한 경우
Azure 자격 증명으로 작업 영역에 로그인합니다.



- 선택 **작업 영역** 왼쪽 메뉴에서 선택한 다음 **사용자** 및 사용자 계정 (이메일 주소)를 선택한
다음 사용자 작업 공간의 상단에 있는 아래쪽 화살표를 선택하고 **Import(가져오기)**를
선택합니다.



- 전자 필기장 가져 오기 대화 상자에서 가져올 대상 **URL**을 선택한 다음 상자에 다음을
붙여 넣습니다.

<https://github.com/azure-datasolution/CloudScaleAnalytics/blob/master/Hands%20on%20Lab/Day2/lab-files/ScaleAnalyticsdbc>

IMPORT NOTEBOOKS

Import from: File URL

Accepted formats: .dbc, .scala, .py, .sql, .r, .ipynb, .Rmd, .html

(To import a library, such as a jar or egg, [click here](#))

4. **Import(가져 오기)**를 선택하십시오 .
5. 이제 사용자 작업 공간에 **ScaleAnalytics** 라는 폴더가 나타납니다 . 이 폴더에는이 실습에서 사용할 모든 노트북이 들어 있습니다.

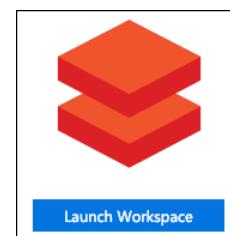
작업 7 : Azure Databricks 주요 자격 증명 지원 비밀 구성

이 작업에서는 Azure Databricks 작업 영역에 연결하고 Azure Key Vault 계정을 백업 저장소로 사용하도록 Azure Databricks 비밀을 구성합니다.

1. [Azure Portal](#)로 돌아가서 새로 프로비저닝 된 Key Vault 계정으로 이동 한 다음 왼쪽 메뉴에서 **속성** 을 선택 합니다.
2. 복사 **DNS 이름 및 자원 ID** 속성 값을 메모장 또는 나중에 참조 할 수있는 몇 가지 다른 텍스트 응용 프로그램에 붙여 넣습니다. 이 값들은 다음 섹션에서 사용됩니다.

The screenshot shows the 'woodgrove-vault - Properties' blade in the Azure portal. On the left, there's a sidebar with 'Settings' and several options like 'Keys', 'Secrets', 'Certificates', 'Access policies', 'Firewalls and virtual networks', and 'Properties'. The 'Properties' option is highlighted with a red box. On the right, there are sections for 'NAME' (woodgrove-vault), 'SKU (PRICING TIER)' (Standard), and two fields highlighted with a red box: 'DNS NAME' (https://woodgrove-vault.vault.azure.net/) and 'RESOURCE ID' (/subscriptions/.../resourcegroups/hands-on-lab/pr...). Below these fields is a 'LOCATION' section.

3. 위에서 프로비저닝 한 Azure Databricks 작업 영역으로 이동 하고 개요 블레이드에서 **작업 영역 시작**을 선택 하고 필요한 경우 Azure 자격 증명으로 작업 영역에 로그인합니다.



4. 브라우저의 URL 표시 줄에서 **#secrets/createScope** 를 Azure Databricks 기본 URL에 추가하십시오.
(예 : <https://eastus.azuredatabricks.net#secrets/createScope>)
5. 이름에 **key-vault-secrets** 입력하십시오 .
6. Manage Principal 드롭 다운에서 **Creator** 선택 지정합니다.

관리 권한을 통해 사용자는이 비밀 범위를 읽고 쓸 수 있으며 Azure Databricks Premium Plan 의 계정 인 경우 범위에 대한 권한을 변경할 수 있습니다.

Creator 를 선택할 수 있으려면 계정에 Azure Databricks Premium Plan 이 있어야합니다. 이 방법은 권장되는 방법입니다. 비밀 범위를 만들 때 생성자에게 MANAGE 권한을 부여한 다음 범위를 테스트 한 후 더 세분화 된 액세스 권한을 할당하십시오.

7. Key Vault 생성 단계에서 이전에 복사 한 **DNS 이름과 Resource ID** 를 입력하십시오.

DNS Name	https://woodgrove-vault.vault.azure.net/
Resource ID	/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourcegroups/hands-on-lab/providers/Microsoft.KeyVault/vaults/woodgrove-vault

8. 작성을 선택 하십시오 .

HomePage / Create Secret Scope

Create Secret Scope

A store for secrets that is identified by a name and backed by a specific store type. [Learn more](#)

Scope Name ?

Manage Principal ?

Creator

Azure Key Vault ?

DNS Name

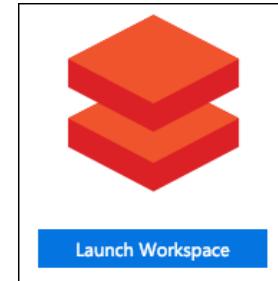
Resource ID

잠시 후 비밀 범위가 만들어 졌음을 확인하는 대화 상자가 나타납니다.

작업 8 : Databricks 에 Azure Cosmos DB Spark 커넥터 및 scikit-learn 라이브러리 설치

이 작업에서는 [Azure Cosmos DB Spark 커넥터](#) 및 scikit-learn 라이브러리를 Databricks 클러스터에 설치합니다. Cosmos DB 커넥터를 사용하면 Apache Spark DataFrames 를 통해 Azure Cosmos DB 에서 쉽게 읽고 쓸 수 있습니다.

1. [Azure Portal](#) 에서 Azure Databricks 작업 영역으로 이동 하고
개요 블레이드에서 **작업 영역 시작**을 선택하고 필요한 경우
Azure 자격 증명으로 작업 영역에 로그인합니다.



2. **Workspace** 를 선택, 왼쪽 메뉴에서 **shared(공유)** 옆에 있는 drop-down 선택하고
create(만들기) 선택하고 다음은 **Library** 선택합니다..

A screenshot of the Microsoft Azure Databricks workspace interface. On the left, there is a sidebar with icons for Azure Databricks, Home, Workspace (which is highlighted with a red box), Recents, Data, and a plus sign icon. The main area is titled 'Workspace' and shows a list of shared items: Documentation, Release Notes, Training & Tutorials, and Shared. Under 'Shared', there is a 'Create' dropdown menu with options: Notebook, Library (which is highlighted with a red box), Folder, Clone, Import, and Export. To the right, there are sections for Permissions and Mounts.

3. 라이브러리 생성 페이지의 라이브러리 소스에서 **Maven** 을 선택한 다음 좌표 텍스트 상자
옆의 **패키지 검색** 을 선택하십시오 .

Create Library

Library Source

- [Upload](#)
- [DBFS](#)
- [PyPI](#)
- [Maven](#)
- [CRAN](#)

Repository [?](#)

Optional

Coordinates

Maven Coordinates (com.databricks:spark-csv_2.10:1.0.0)

[Search Packages](#)

Exclusions

Dependencies to exclude (log4j:log4j,junit:junit)

[Create](#) [Cancel](#)

4. 검색 패키지 대화 상자 의 소스 드롭 다운에서 **Maven Central** 을 선택 하고 검색 상자에 **azure-cosmosdb-spark** 를 입력 한 다음 **azure-cosmosdb-spark_2.4.0_2.11 release 1.5.0 선택을 클릭 합니다.**

Search Packages

Maven Central	Q azure-cosmosdb-spark		
Group Id	Artifact Id	Releases	Options
com.microsoft.azure	azure-cosmosdb-spark_2.4.0_2.11	1.3.5	Select
com.microsoft.azure	azure-cosmosdb-spark_2.1.0_2.11	1.2.6	Select
com.microsoft.azure	azure-cosmosdb-spark_2.2.0_2.11	1.1.1	Select
com.microsoft.azure	azure-cosmosdb-spark_2.2.0_2.10	1.0.0	Select

5. 라이브러리 설치를 마치 려면 **작성을 선택 하십시오 .**

Create Library

Library Source

- [Upload](#)
- [DBFS](#)
- [PyPI](#)
- [Maven](#)
- [CRAN](#)

Repository [?](#)

Optional

Coordinates

com.microsoft.azure:azure-cosmosdb-spark_2.4.0_2.11:1.3.5

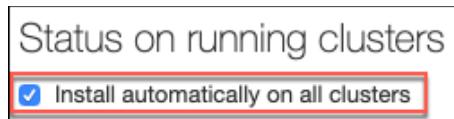
[Search Packages](#)

Exclusions

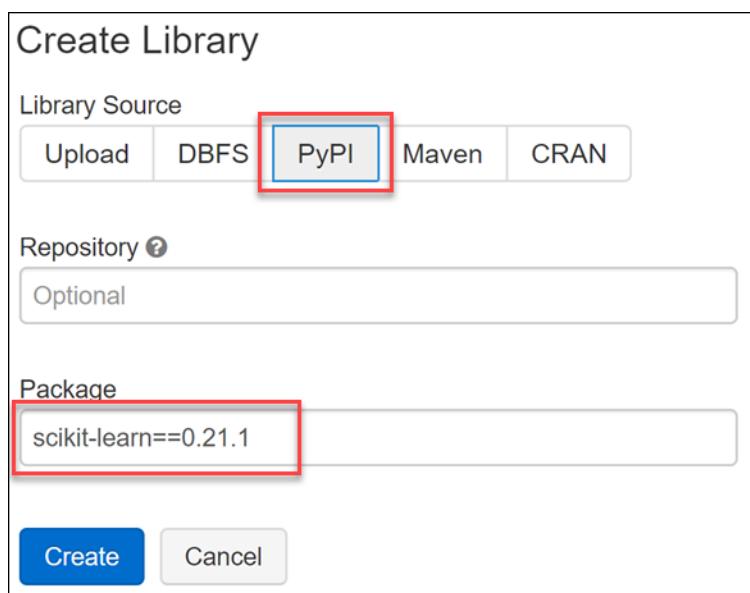
Dependencies to exclude (log4j:log4j,junit:junit)

[Create](#) [Cancel](#)

6. 다음 화면에서 모든 클러스터에 자동 설치 상자를 선택하고 프롬프트가 표시되면 확인을 선택하십시오.



7. 작업 공간에서 공유 폴더를 다시 선택하고 컨텍스트 메뉴에서 작성 및 라이브러리를 선택하십시오.
8. 라이브러리 작성 대화 상자에서 PyPI 를 라이브러리 소스로 선택하고 패키지 상자에 scikit-learn == 0.21.1 을 입력 한 후 작성 을 선택 하십시오.

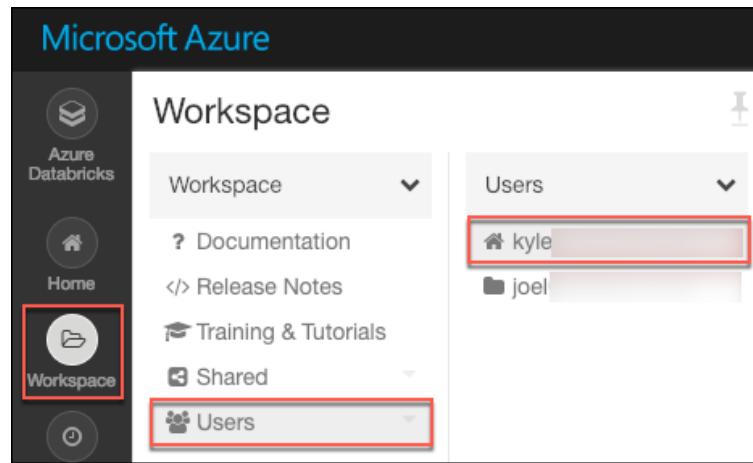


9. 다음 화면에서 모든 클러스터에 자동 설치 확인란을 선택하지 말고 프롬프트가 표시되면 확인을 선택하십시오. 이 라이브러리는 작업 클러스터에 대한 참조로만 필요합니다. 이 scikit-learn 을 Exercise 3 의 실습 클러스터에 직접 추가합니다.

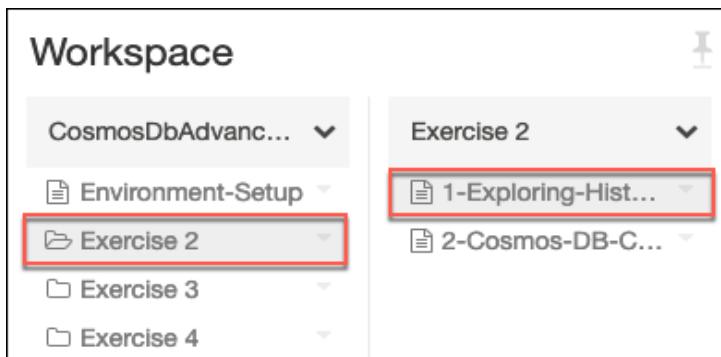
작업 9 : Azure Databricks 및 Spark 를 사용하여 기록 트랜잭션 데이터 탐색

이 작업에서는 Azure Databricks 노트북을 사용하여 기록 트랜잭션 데이터를 다운로드하고 탐색합니다.

- 당신의 Databricks 작업 공간에서, 사용자 및 사용자 계정을 선택하세요.



- 사용자 작업 공간에서 **ScaleAnalytics** 폴더를 선택한 후 **Exercise2** 폴더를 선택하고 이름이 **1-Exploring-Historical-Transactions** 노트북을 선택하십시오.



- [Exercise2] **1-Exploring-Historical-Transactions** 노트북이 작업의 나머지 단계를 완료하기 위한 지시 사항을 따르십시오.

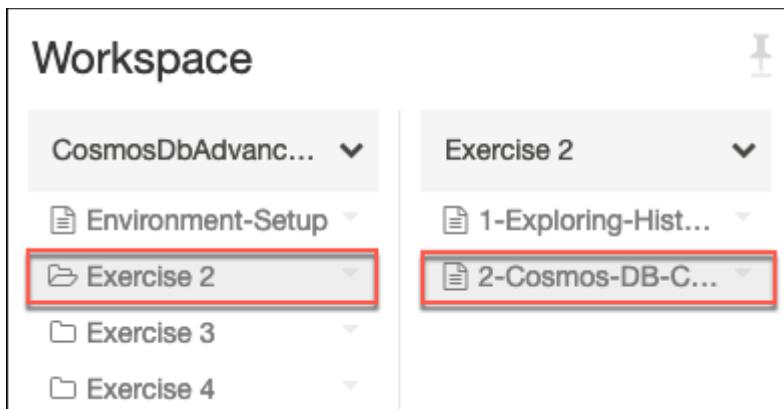
참고 :이 연습에서는 각 노트북의 맨 아래에 다음 작업을 위해 노트북으로 이동하는 링크가 있으므로이 연습을 위해 문서와 Databricks 노트북 간에 앞뒤로 이동할 필요가 없습니다.

작업 10 : Azure Databricks에서 Cosmos DB 변경 피드 및 Spark 구조적 스트리밍을 사용하여 스트리밍 트랜잭션에 응답

이 작업에서는 Azure Databricks 노트북을 사용하여 Azure Databricks 노트북에서 Cosmos DB 인스턴스에 대한 연결을 만들고 Cosmos DB 변경 피드에서 스트리밍 데이터를 쿼리합니다.

- 당신의 Databricks 작업 공간에서, 선택 **작업**을 한 후, 왼쪽 메뉴에서 선택 **사용자 및 사용자 계정**을.

2. 사용자 작업 공간에서 **ScaleAnalytics** 폴더를 선택한 후 **Exercise 2** 폴더를 선택하고 이름이 **2-Cosmos-DB-Change-Feed** 노트북을 선택하십시오 .



3. **2-Cosmos-DB-Change-Feed** 노트북에서 ,이 작업의 나머지 단계를 완료하는 지침을 따르십시오.

연습 3 : 사기 모델 작성 및 평가

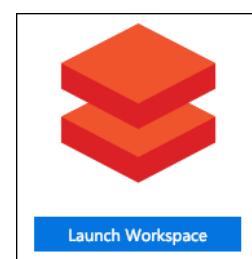
[소요 시간 : 45 분]

이 연습에서는 웹 프런트 엔드에서 발생하는 트랜잭션의 실시간 스코어링에 사용되는 사기 모델을 만들고 평가합니다. 목표는 사기 거래가 처리되기 전에 차단하는 것입니다. 그런 다음 실습 4에서 발생하는 일괄 처리 중에 실행되는 의심스러운 트랜잭션을 탐지하기 위한 모델을 만듭니다. 마지막으로, 사기성 트랜잭션 모델을 배포하고 Databricks 노트북 내에서 HTTP REST 호출을 통해 테스트합니다.

작업 1 : Databricks에 AzureML 및 Scikit-Learn 라이브러리 설치

이 작업에서는 Databricks 클러스터에 필수 **AzureML** 및 **Scikit-Learn** 라이브러리를 설치합니다. 이 라이브러리는 기계 학습 모델을 교육하고 배포 할 때 사용됩니다. 표시된 순서대로 설치하는 것이 중요합니다.

1. [Azure Portal](#)에서 Azure Databricks 작업 영역으로 이동하고 개요 블레이드에서 **작업 영역 시작**을 선택하고 필요한 경우 Azure 자격 증명으로 작업 영역에 로그인합니다.



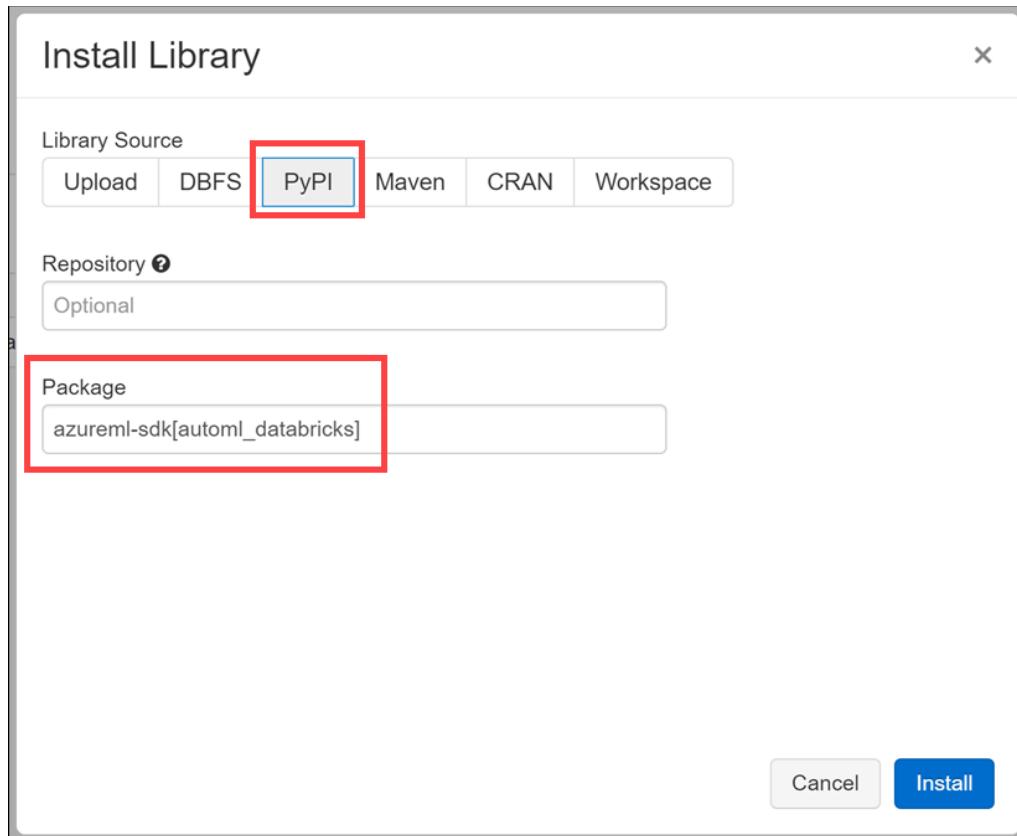
2. 왼쪽 목록에서 클러스터를 선택합니다.

Name	State	Nodes
lab-cluster	Running	3

3. 라이브러리에 설치된 라이브러리 목록을 표시 하는 라이브러리 탭을 선택하십시오 . Azure Cosmos DB Spark 커넥터가 설치되어 있어야합니다. 라이브러리 목록 위에서 새로 설치를 선택하십시오 .

Name	Type	Status
azure_cosmosdb_spark_2_3_0_2_11_1_2_2_u...	JAR	Insta

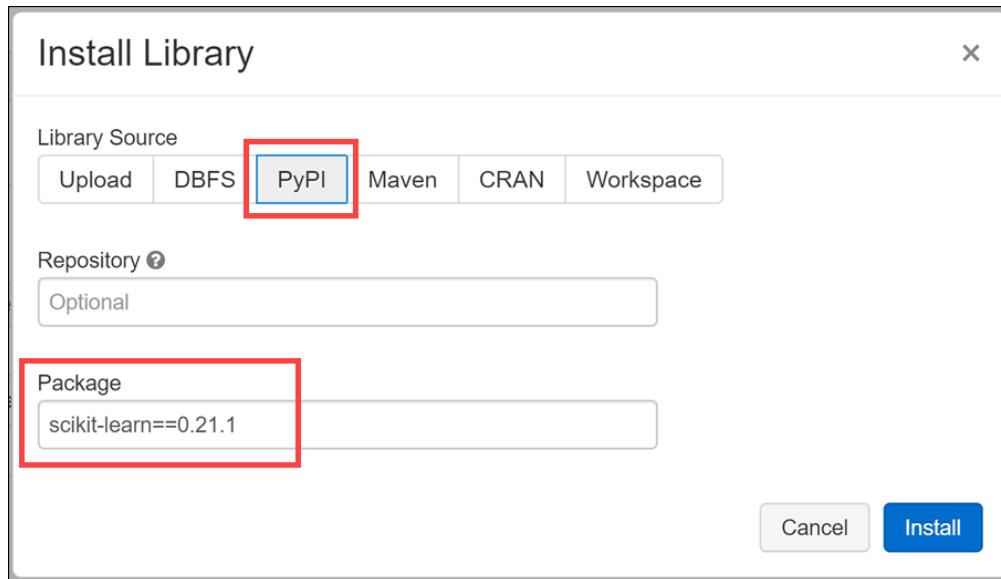
4. 대화 상자가 나타나면 라이브러리 소스로 PyPI를 선택하십시오. 입력에서 패키지를 선택한 다음 azureml-sdk[automl_databricks] 설치합니다.



5. 중요 : AzureML 라이브러리의 상태가 **Installed**로 표시될 때까지 기다리십시오. Scikit-Learn을 설치하기 전에 완료해야합니다.

Configuration Notebooks (0) Libraries (2) Event Log Spark UI Driver Logs Spark					
		Uninstall		Install New	
	Name	Type	Status	S	
	azure_cosmosdb_spark_2_3_0_2_11_1_2_2_u...	JAR	Installed		
	azureml-sdk[automl_databricks]	PyPI	Installed		

6. 새로 설치를 다시 선택 하십시오.
7. 대화 상자가 나타나면 라이브러리 소스에서 PyPI를 선택하십시오. 입력에서 패키지를 선택한 다음 scikit-learn==0.21.1를 설치합니다.



8. Scikit-Learn 라이브러리가 설치되면 라이브러리 목록은 다음과 같아야합니다.

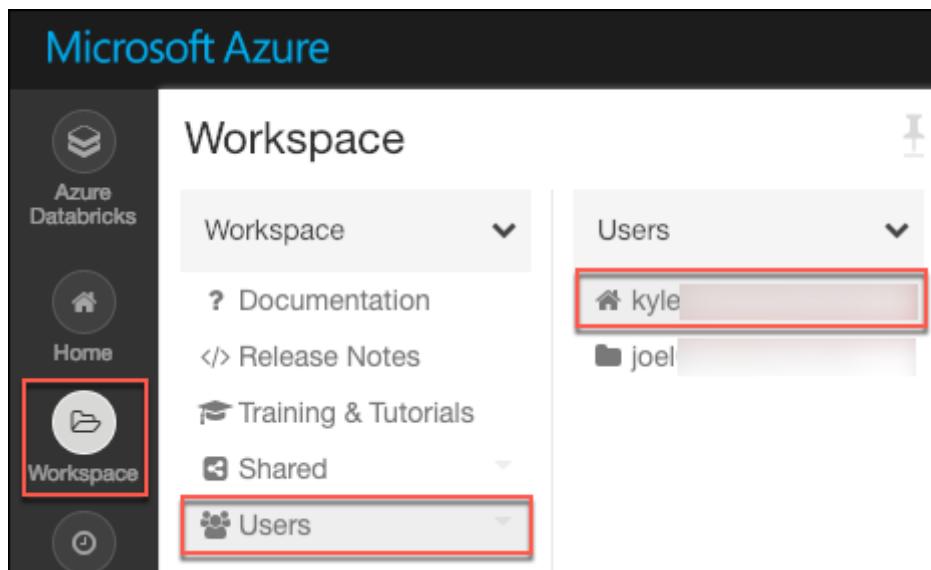
	Name	Type	Status
<input type="checkbox"/>	azureml-sdk[automl_databricks]	PyPI	Installed
<input type="checkbox"/>	com.microsoft.azure:azure-cosmosdb-spark_2....	Maven	Installed
<input type="checkbox"/>	scikit-learn==0.21.1	PyPI	Installed

작업 2 : 스코어링 웹 서비스 준비 및 배포

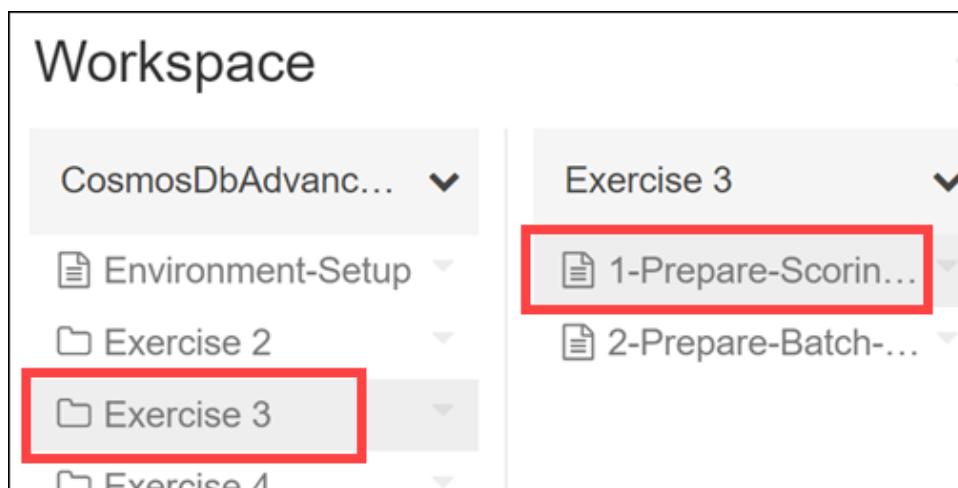
이 작업에서는 Azure Databricks 노트북을 사용하여 트랜잭션 및 계정 데이터를 탐색합니다. 또한 데이터 정리를 수행하고 데이터가 스코어링을 위해 모델로 전달 될 때마다 이러한 변환을

적용하는 기능 엔지니어링 파이프 라인을 작성합니다. 마지막으로 사기 거래를 탐지하는 기계 학습 모델을 교육하고 배포합니다.

- 당신의 Databricks 작업 공간에서, 왼쪽 메뉴에서 **Workspace** 를 선택 한 후, **사용자 선택** 및 사용자 계정을 선택합니다.



- 사용자 작업 공간에서 **ScaleAnalytics** 폴더를 선택한 후 **Exercise 3** 폴더를 선택하고 **1-Prepare-Scoring-Web-Service** 라는 노트북을 선택하십시오 .



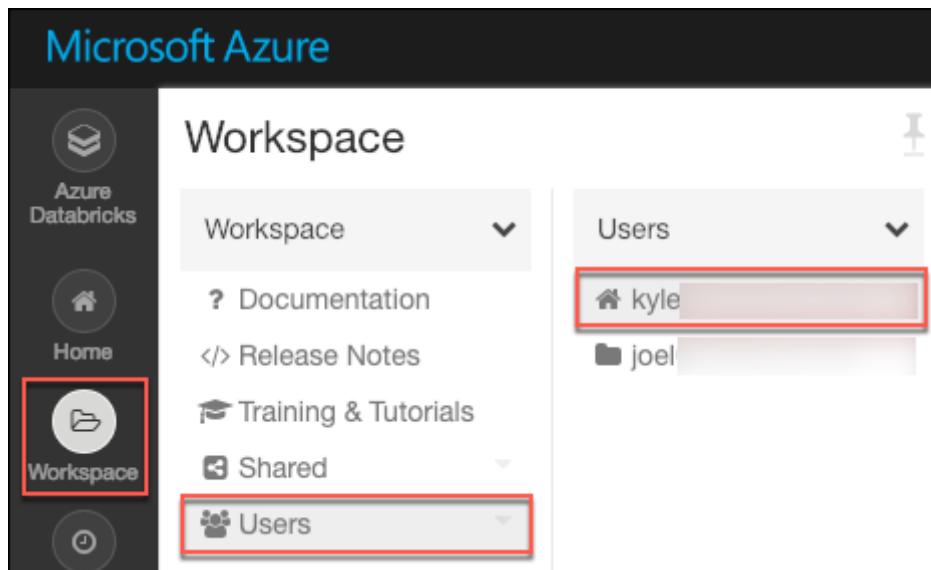
- 1-Prepare-Scoring-Web-Service** 노트북에서 ,이 작업의 나머지 단계를 완료하기위한 지시 사항을 따르십시오.

참고 :이 연습에서는 각 노트북의 맨 아래에 다음 작업을 위해 노트북으로 이동하는 링크가 있으므로이 연습을 위해이 문서와 Databricks 노트북간에 앞뒤로 이동할 필요가 없습니다.

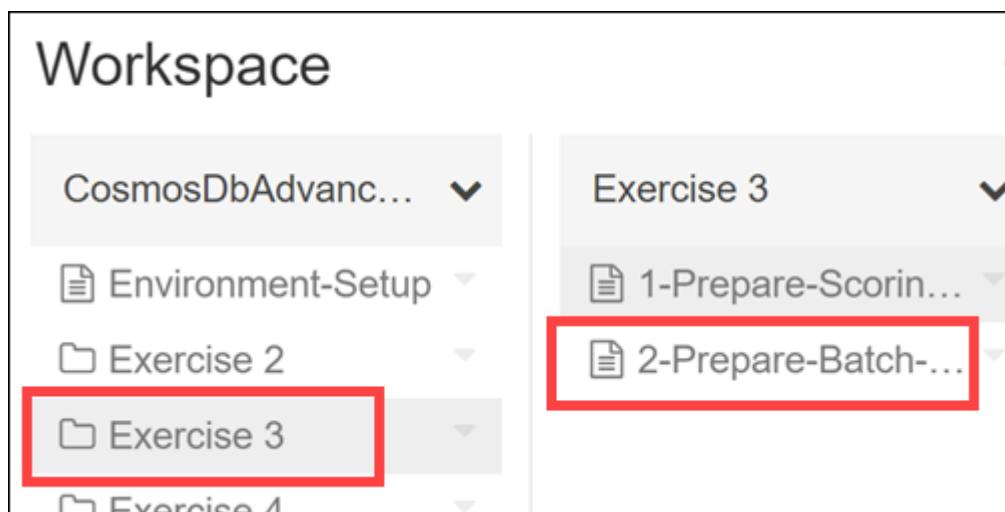
작업 3 : 배치 스코어링 모델 준비

이 작업에서는 Azure Databricks 노트북을 사용하여 배치 스코어링에 사용될 의심스러운 활동을 감지하는 데 사용되는 모델을 준비합니다.

1. 당신의 Databricks 작업 공간에서, 선택 **작업**을 한 후, 왼쪽 메뉴에서 선택 **사용자 및 사용자 계정**을.



2. 사용자 작업 공간에서 **ScaleAnalytics** 폴더를 선택한 후 **Exercise 3** 폴더를 선택하고 이름이 **2-Prepare-Batch-Scoring-Model** 인 노트북을 선택하십시오 .



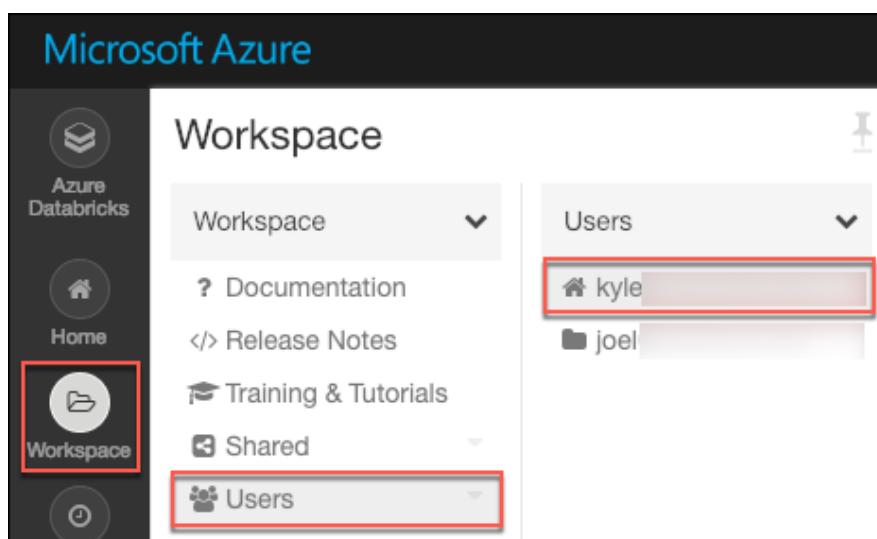
3. **2-Prepare-Batch-Scoring-Model** 노트북에서 이 작업의 나머지 단계를 완료하기 위한 지시 사항을 따르십시오.

연습 4 : 배치점수 데이터 배포

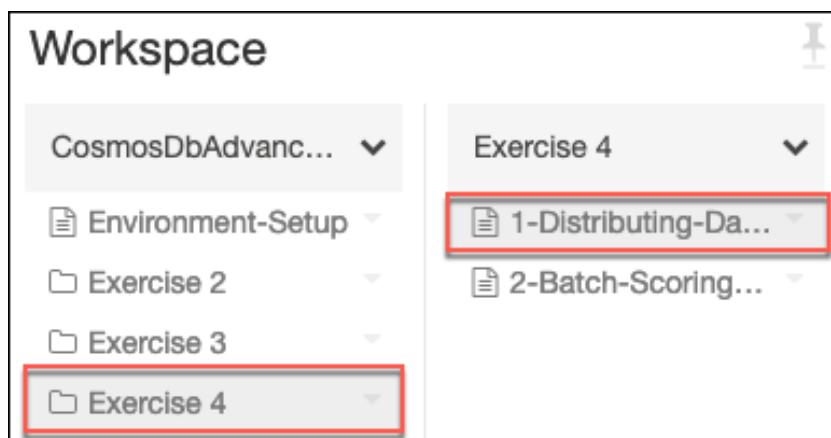
작업 1 : Cosmos DB 를 사용하여 배치 점수 데이터 배포

이 작업에서는 Azure Databricks 노트북을 `transactions` 사용하여 머신 러닝 모델로 Databricks 델타 테이블에 저장된 데이터를 일괄 처리 합니다. 스코어링 결과는 새로운 `scored_transactions` 델타 테이블에 기록되며 의심스러운 거래는 Cosmos DB 에도 다시 기록됩니다.

- 당신의 Databricks 작업 공간에 왼쪽 메뉴에서 Workspace 선택 후, 사용자 및 사용자 계정을 선택.



- 사용자 작업 공간에서 **ScaleAnalytics** 폴더를 선택한 다음 **Exercise 4** 폴더를 선택하고 **1-Distributing-Data-Globally** 라는 노트북을 선택하십시오 .

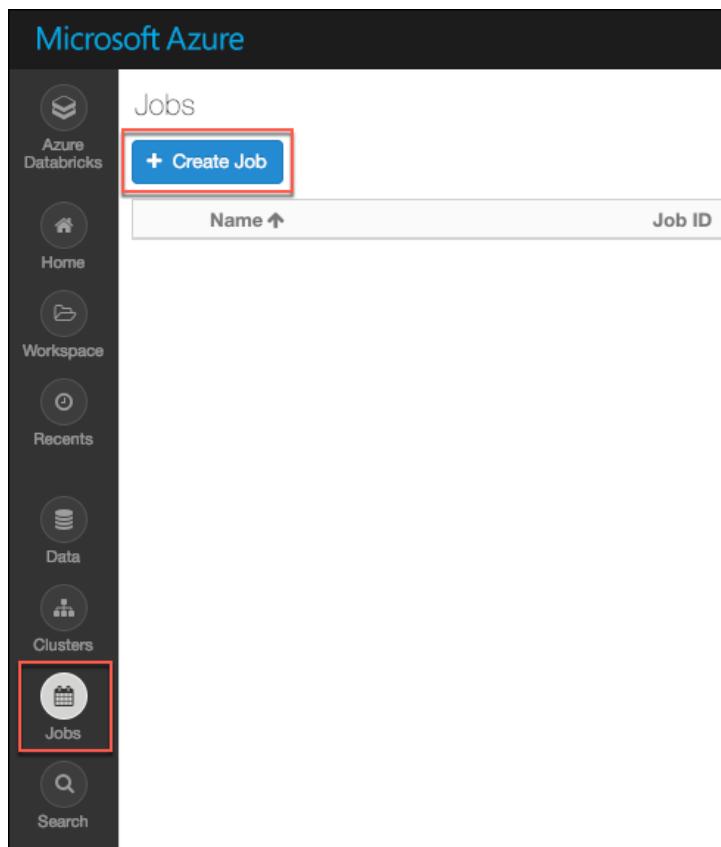


3. **1-Distributing-Data-Globally** 노트북에서 이 작업의 나머지 단계를 완료하기 위한 지시 사항을 따르십시오.

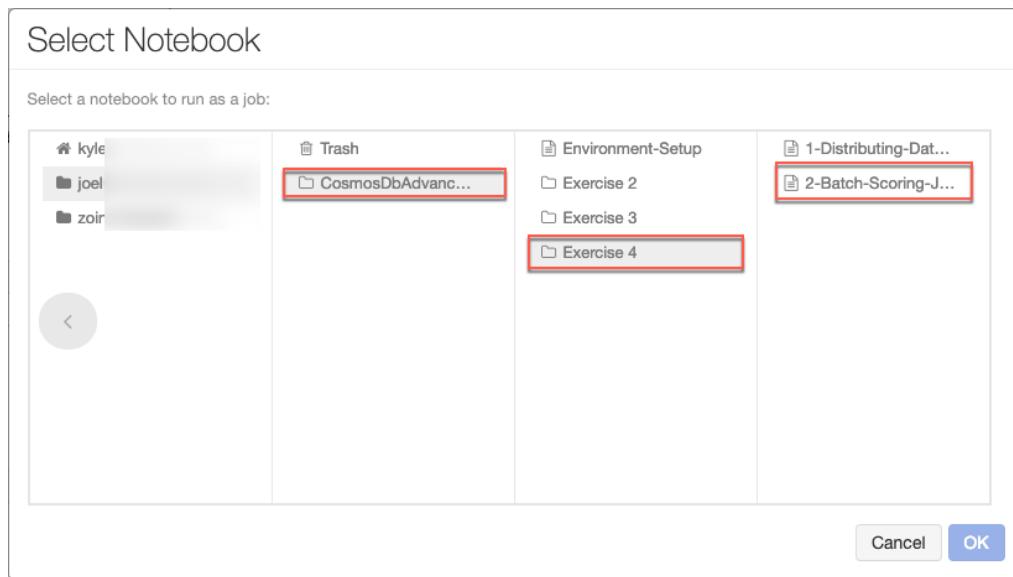
작업 2 : Azure Databricks 작업을 사용하여 일정에 따라 트랜잭션 점수를 일괄 처리

이 작업에서는 Azure Databricks 작업을 생성합니다. 이 작업은 시간별 일정에 따라 트랜잭션에 대해 일괄 평가를 수행하는 노트북을 실행합니다.

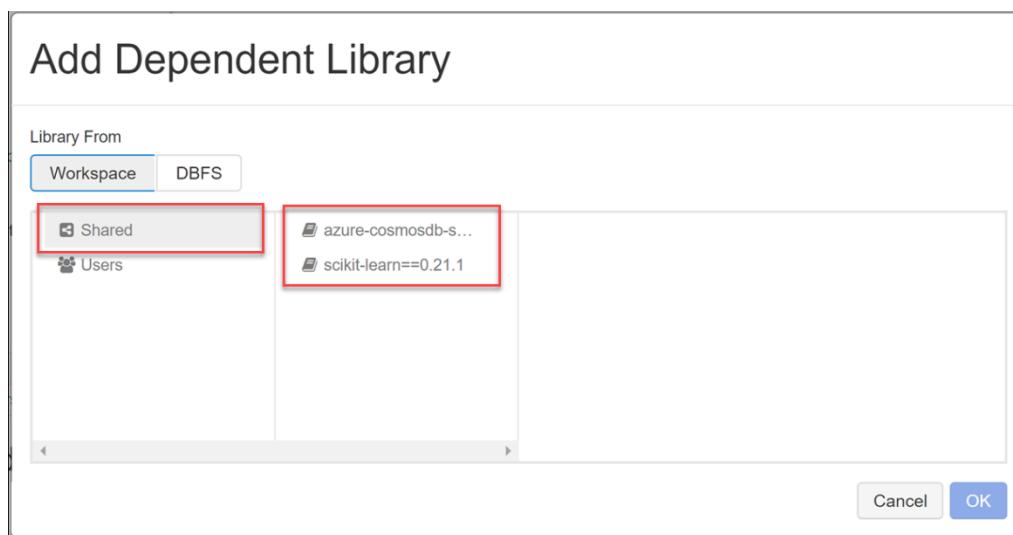
1. 당신의 Databricks 작업 공간으로 이동 후 왼쪽 메뉴에 **Jobs** 선택한 다음 **+ 선택**하여 작업을 만듭니다 .



2. 제목없는 작업 화면에서 다음 단계를 완료하십시오.
 - o **Batch-Scoring-Job** 과 같은 제목을 입력하십시오 .
 - o **Task** 옆의 **노트북 선택** 링크를 선택하고 **노트북 선택** 대화 상자에서 **사용자 -> 사용자 계정-> ScaleAnalytics-> Exercise 4** 를 선택한 다음 **2-Batch-Scoring-Job** 노트북을 선택하고 **확인을 선택**하십시오 .



- 종속 라이브러리(Dependent Libraries) 옆에있는 **추가를** 선택하고 공유 폴더로 이동한 다음 **azure-cosmosdb-spark** 라이브러리를 선택하고 **확인을** 선택하십시오 .
- **scikit-learn == 0.21.1** 라이브러리도 추가하려면 위 단계를 반복하십시오 .



- 클러스터 편집을 선택합니다 :
 - **Databricks 런타임 버전** : 런타임 5.5LTS (Scala 2.11, Spark 2.4.3)
 - **파이썬 버전** : 3
 - **Worker Type** : Standard_DS3_v2
 - **Workers**: 1
 - **Driver Type** : Worker 와 동일
 - 고급을 `spark.databricks.delta.preview.enabled true` 펼치고 Spark 구성 상자에 입력되어 있는지 확인하십시오 .

Configure Cluster

Cancel Confirm 8 Workers: 224.0 GB Memory, 64 Cores, 12 DBU
1 Driver: 28.0 GB Memory, 8 Cores, 1.5 DBU Cost \$0.00

Cluster Type
New Cluster

Databricks Runtime Version [Learn more](#)
Runtime: 5.2 (Scala 2.11, Spark 2.4.0)

Python Version
3 New The default Python version for clusters was changed from 2.7 to 3.0.

Autopilot Options
 Enable autoscaling

Worker Type
Standard_DS4_v2 28.0 GB Memory, 8 Cores, 1.5 DBU Workers 8

Driver Type
Same as worker 28.0 GB Memory, 8 Cores, 1.5 DBU

▼ Advanced Options

Spark Tags Logging Init Scripts

Spark Config [?](#)
spark.databricks.delta.preview.enabled true

- 클러스터 구성을 저장하려면 **확인을 선택하십시오**.
- 일정(Schedule) 옆에있는 편집을 선택하고 작업 예약 대화 상자에서 일정을 현재 시간에 가까운 값에서 시작하여 매시간으로 설정하면 적절한 시간에 트리거 된 것을 볼 수 있습니다. 시간대를 선택하고 **확인을 선택하십시오**.

Schedule Job

Schedule
Every hour starting at 00:55 US/Eastern Show Cron Syntax

Cancel Confirm

3. 최종 작업 화면은 다음과 같아야합니다.

Batch-Scoring-Job

Job ID: 1

Task: Notebook at /Users/joel@solliance.net/CosmosDbAdvancedAnalytics/Exercise 4/2-Batch-Scoring-Job - [Edit](#) / [Remove](#)

- ▶ Parameters: [Edit](#)
- Dependent Libraries: [Add](#)
 - com.microsoft.azure:azure-cosmosdb-spark_2.4.0_2.11:1.4.0 - (Maven) [Remove](#)
 - scikit-learn==0.21.1 - (Pypi) [Remove](#)

Cluster: Driver: Standard_DS4_v2, Workers: Standard_DS4_v2, 8 workers, 5.2 (includes Apache Spark 2.4.0, Scala 2.11) [Edit](#)

Schedule: Every hour starting at 12:05am (US/Pacific) [Edit](#) / [Remove](#)

Advanced ▶

4. 완료되면 작업 목록으로 돌아가 려면 <모든 작업>을 선택하십시오.
5. 작업 시작을 기다리는 동안 왼쪽 메뉴에서 **작업 공간**을 선택 **2-Batch-Scoring-Job**하고 Exercise 4 폴더 아래의 노트북으로 이동하십시오.
6. 노트북을 열고 몇 분 동안 배치 스코어링 프로세스를 수행하는 데 사용되는 단계를 이해하십시오. 보시다시피, 이전 작업에서 트랜잭션 데이터를 준비하고 변환하는 과정과 거의 동일합니다.
7. Databricks의 왼쪽 메뉴에서 **클러스터**를 선택한 후 작업 클러스터에 대한 **작업 실행**을 선택하여 작업 진행 상황을 모니터 할 수 있습니다. 노트북이 표시되고 노트북 내에서 실행 시간과 결과를 볼 수 있습니다.

Name	State	Nodes	Driver	Worker	Runtime	Creator	Actions
lab-cluster	Running	3	Standard_DS4_v2	Standard_DS4_v2	5.1 (includes Apa...)	[redacted]	5 3
job-1-run-1	Terminating	9	Standard_DS4_v2	Standard_DS4_v2	5.1 (includes Apa...)	[redacted]	Job Run / Spark UI / Logs [redacted]

연습 5 :보고서

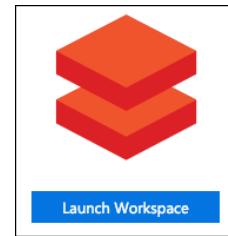
[소요 시간 : 30 분]

이 연습에서는 비즈니스 분석가가 사용할 수 있도록 Power BI에서 대시 보드 및 보고서를 만들고 데이터 과학자와 분석가가 데이터를 대화 형으로 쿼리하고 시각화 할 수 있도록 Azure Databricks 내에서 생성합니다.

작업 1 : 전 세계 사기 추세를 요약하고 시각화하기 위해 Power BI 활용

이 작업에서는 Azure Databricks 클러스터의 JDBC URL을 사용하여 Power BI 데스크톱에서 연결합니다. 그런 다음 보고서를 작성하고 대시 보드에 추가하여 글로벌 사기 경향을 요약하고 시각화하여 데이터에 대한 통찰력을 얻습니다.

1. [Azure Portal](#)에서 Azure Databricks 작업 영역으로 이동하고 개요 블레이드에서 **작업 영역 시작**을 선택하고 필요한 경우 Azure 자격 증명으로 작업 영역에 로그인합니다.



2. 선택 클러스터 다음 대화 형 클러스터의 목록에서 클러스터를 선택, 왼쪽 메뉴에서를.

Microsoft Azure

Clusters

+ Create Cluster

Interactive Clusters

Name	State	Nodes
lab-cluster	Running	3

Job Clusters

3. 아래로 스크롤하여 고급 옵션 섹션을 펼친 다음 JDBC / ODBC 탭을 선택 하십시오.
첫 번째 JDBC URL 값을 복사하십시오 .

lab-cluster

Edit Clone Restart Terminate Delete

Configuration Notebooks (3) Libraries (4) Event Log Spark UI Driver Logs Spark Cluster UI - Master

Driver Type
Standard_DS4_v2 28.0 GB Memory, 8 Cores, 1.5 DBU

Advanced Options

Spark Tags Logging Init Scripts JDBC/ODBC Permissions

Server Hostname
eastus.azuredatabricks.net

Port
443

Protocol
HTTPS

HTTP Path
sql/protocolv1/o/8433778235244215/0210-035431-quad242 (unique)
sql/protocolv1/o/8433778235244215/lab-cluster (alias, not guaranteed unique)

JDBC URL

```
jdbc:spark://eastus.azuredatabricks.net:443/default;transportMode=http;ssl=1;httpPath=/sql/protocolv1/o/8433778235244215/0210-035431-quad242;AuthMech=3;UID=token;PWD=<personal-access-token>
```

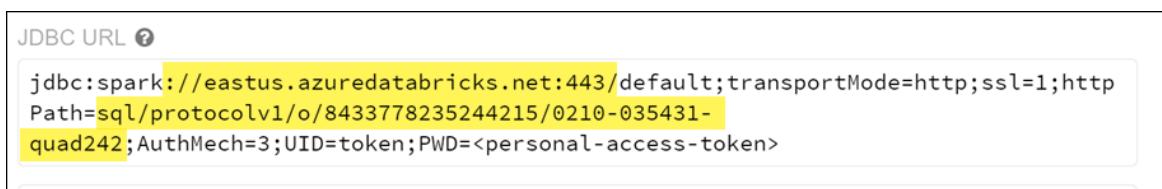
```
jdbc:spark://eastus.azuredatabricks.net:443/default;transportMode=http;ssl=1;httpPath=/sql/protocolv1/o/8433778235244215/lab-cluster;AuthMech=3;UID=token;PWD=<personal-access-token>
```

4. 이제 Power BI Desktop에서 Spark 클러스터 연결을 설정하는 데 사용할 JDBC 서버 주소를 구성하도록 JDBC URL을 수정해야합니다.

o JDBC URL에서 :

- https를 jdbc:spark로 교체 합니다.
- 포트 번호와 /sql 사이의 경로에 모든 것을 제거하십시오 .
- URL 끝에서 ;AuthMech=3;UID=token;PWD=<personal-access-token> 문자열을 제거하십시오 .

아래 이미지에서 강조 표시로 표시된 구성 요소를 유지하십시오.

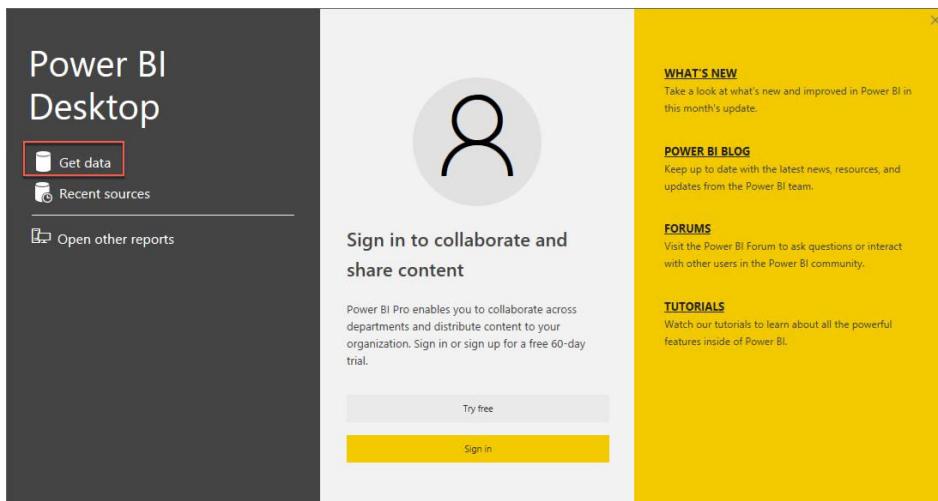


o 이 예에서 서버 주소는 다음과 같습니다.

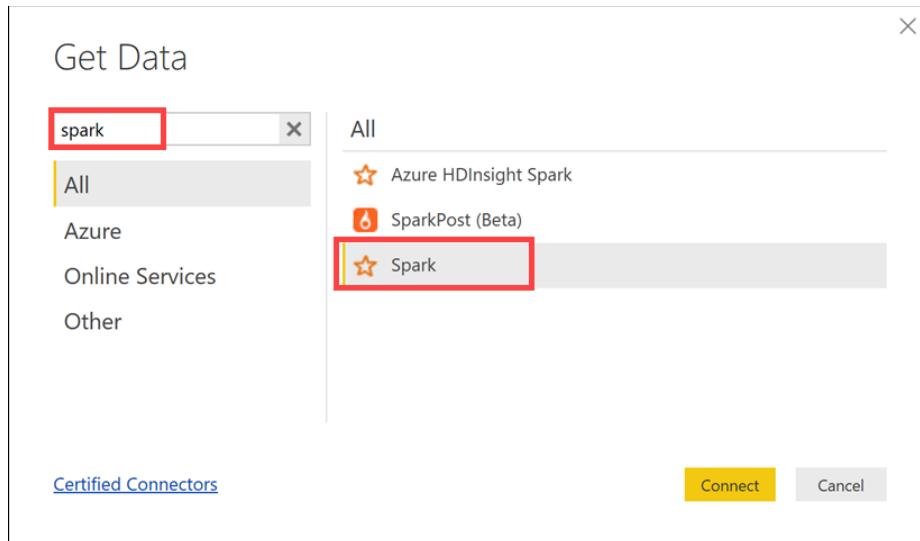
<https://eastus.azuredatabricks.net:443/sql/protocolv1/o/8433778235244215/0210-035431-quad242>

o 서버 주소를 복사하십시오.

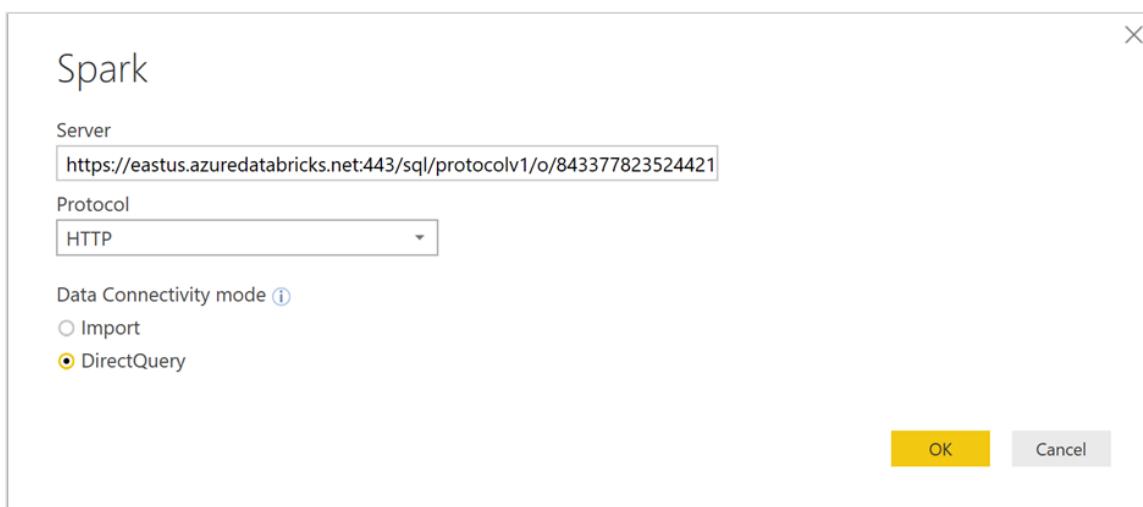
5. Power BI Desktop을 연 다음 데이터 가져 오기 를 선택 합니다 .



6. 대화상자에서 데이터 가져 오기를 선택 후에 **spark**를 넣은 결과 목록에서 **Spark**를 선택하십시오 .

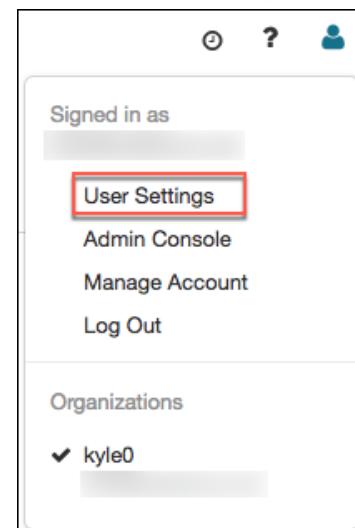


7. 위에서 작성한 Databricks 서버 주소를 서버 필드에 입력하십시오.
8. 프로토콜을 HTTP로 설정하십시오.
9. 데이터 연결 모드로 DirectQuery를 선택한 다음 확인을 선택하십시오.

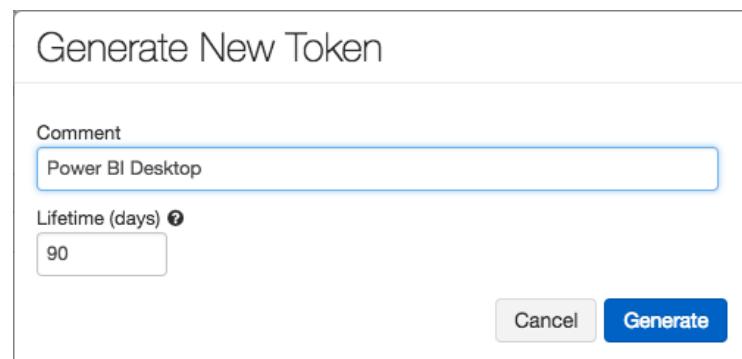


데이터 연결 모드를 DirectQuery로 설정하면 처리를 Spark로 오프로드 할 수 있습니다. 이는 대량의 데이터가 있거나 거의 실시간 분석을 원할 때 이상적입니다.

10. 다음 화면에서 자격 증명을 입력하기 전에 Azure Databricks에서 액세스 토큰을 만들어야합니다. Databricks 작업 공간의 오른쪽 상단에서 계정 아이콘을 선택한 다음 메뉴에서 사용자 설정을 선택하십시오.

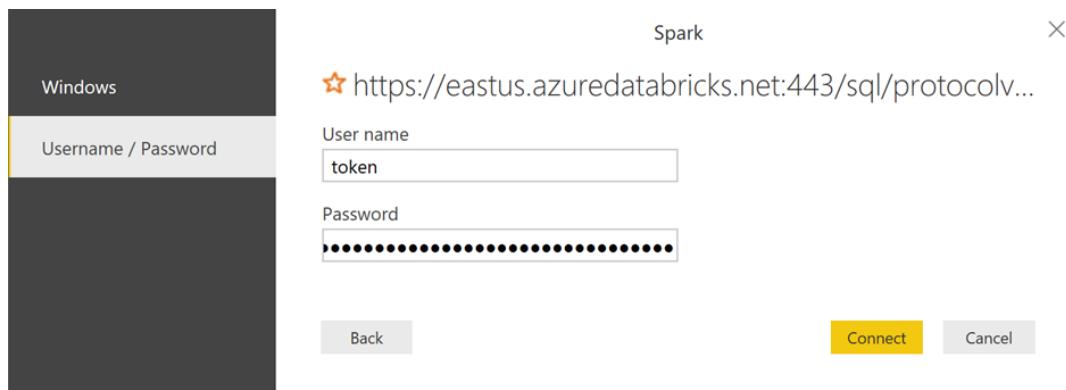


11. 사용자 설정 페이지에서 새 토큰 생성을 선택하고 주석에 "Power BI Desktop"을 입력 한 다음 생성을 선택하십시오.



12. 생성 된 토큰을 복사하여 아래에 두 번 이상 필요할 수 있으므로 저장하십시오. 참고 : 토큰 생성 대화 상자를 닫으면 Databricks에서 토큰에 액세스 할 수 없으므로이 값을 텍스트 랩 또는이 실습 중에 액세스 할 수있는 다른 위치에 저장하십시오.

13. Power BI Desktop으로 돌아가서 사용자 이름으로 "토큰"을 입력하고 Databricks에서 복사한 액세스 토큰을 암호 필드에 붙여 넣습니다.



14. `scored_transactions` 및 `percent_suspicious` 을 선택하신 후 **로드** 를 클릭하세요.

ipCountryCode	SuspiciousTransactionCount	TotalTransactionCount	PercentSuspicious
co	0	187	
fi	0	741	
mt	0	24	
mv	0	2	
pm	0	3	
ug	0	5	
bz	0	9	
sr	0	5	
sv	0	1876	
ad	0	1	
bh	0	25	
gy	0	3	
hu	0	71	
is	0	37	
jp	96	1913	
lr	0	2	
mc	0	10	
gt	0	24	
lb	0	9	
md	0	3	
tn	0	5	
uy	0	40	
by	0	12	

Load **Edit** **Cancel**

15. 잠시 후 표가 오른쪽에 나열된 빈
보고서 화면으로
리디렉션됩니다. 오른쪽 메뉴에서
시각화 아래 및 테이블 및 필드 목록
옆의 도넛 형
차트 시각화를 선택 합니다.

VISUALIZATIONS

FIELDS

Legend: ipCountryCode

Values: transactionAmountUSD

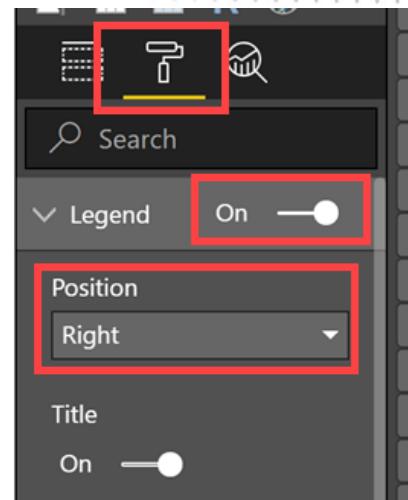
FIELDS:

- percent_suspicious
- scored_transactions
 - accountAge
 - accountCountry
 - accountId
 - accountPostalCode
 - accountState
 - browserLanguage
 - cardType
 - cvvVerifyResult
 - digitalItemCount
 - ipCountryCode
 - ipPostcode
 - ipState
 - isProxyIP
 - isSuspicious
 - isUserRegistered
 - localHour
 - numPaymentRejects1d...
 - paymentBillingCountry...
 - paymentBillingPostalC...
 - paymentBillingState
 - paymentInstrumentAg...
 - paymentInstrumentType
 - physicalItemCount
 - transactionAmount
 - transactionAmountUSD
 - transactionCurrencyCo...
 - transactionDateTime

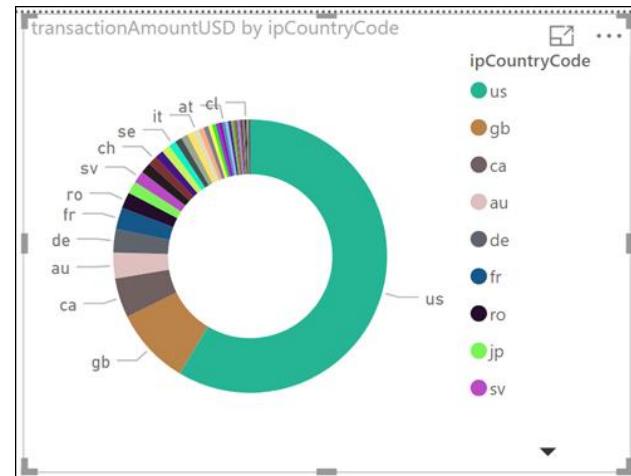
16. 확장 `scored_transactions` 테이블
아래에서, `ipCountryCode` 를
범주(Legend)로 드래그 하고,
그리고 `transactionAmountUSD` 를
마찬가지 값(Values)으로
드래그합니다.

17. 이제 도넛 형

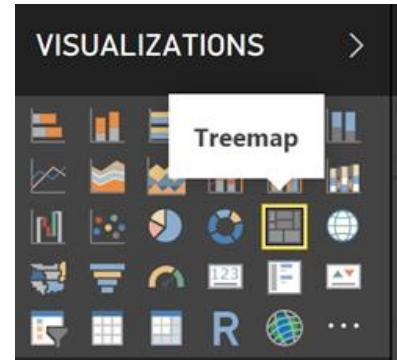
차트의 **형식** 탭을 선택하고 아래의 **범례** 섹션을 확장하십시오. 범례를 켜 려면 **커기**를 선택하고 **위치** 아래에서 **오른쪽**을 선택하십시오. 차트 범례가 켜지고 오른쪽에 표시됩니다.



18. 도넛 형 차트는 다음과 유사하며 국가 코드별로 미국 달러 거래량을 표시합니다.



19. 보고서에서 빈 영역을 선택하여 도넛 형 차트를 선택 해제하십시오. 이제 트리 맵 시각화를 선택하십시오.



20. `scored_transactions` 테이블에서 `transactionAmountUSD` 필드를 그룹으로, `isSuspicious` 를 값으로 드래그 합니다 .

The screenshot shows the Power BI visualizations pane on the left and the Fields pane on the right.

VISUALIZATIONS pane:

- Show/hide pane
- Group (selected)
- transactionAmountUSD (selected)
- Details
- Add data fields here
- Values
- isSuspicious (selected)
- Tooltips
- Add data fields here
- FILTERS
- Visual level filters
- isSuspicious (selected)
- is (All)
- transactionAmountUSD (selected)
- is (All)
- Page level filters
- Drag data fields here
- Report level filters

FIELDS pane:

- Search
- percent_suspicious
- scored_transactions (selected)
- accountAge
- accountCountry
- accountID
- accountPostalCode
- accountState
- browserLanguage
- cardType
- cvvVerifyResult
- digitalItemCount
- ipCountryCode
- ipPostcode
- ipState
- isProxyIP
- isSuspicious (selected)
- isUserRegistered
- localHour
- numPaymentRejects1d...
- paymentBillingCountry...
- paymentBillingPostalC...
- paymentBillingState
- paymentInstrumentAg...
- paymentInstrumentType
- physicalItemCount
- transactionAmount
- transactionAmountUSD (selected)
- transactionCurrencyCo...

21. 보고서에서 빈 영역을 선택하여 트리 맵 차트를 선택

해제하십시오. 이제 맵 시각화를 선택하십시오 . `scored_transactions` 테이블에서 `ipCountryCode`의 필드를 위치(Location)로, `isSuspicious`를 크기로 드래그 합니다 .

The screenshot shows the Power BI interface with two main panes: 'VISUALIZATIONS' on the left and 'FIELDS' on the right.

VISUALIZATIONS Pane:

- Shows various visualization icons.
- Selected: 'Location' (map icon).
- Fields:
 - 'ipCountryCode' is selected and highlighted with a red arrow pointing to it.
 - 'isSuspicious' is selected and highlighted with a red arrow pointing to it.

FIELDS Pane:

- Search bar: 'Search'.
- Table of fields under 'scored_transactions':
 - accountAge
 - accountCountry
 - accountID
 - accountPostalCode
 - accountState
 - browserLanguage
 - cardType
 - cvvVerifyResult
 - digitalItemCount
 - ipCountryCode (selected)
 - ipPostcode
 - ipState
 - isProxyIP
 - isSuspicious (selected)
 - isUserRegistered
 - localHour
 - numPaymentRejects1d...
 - paymentBillingCountry...
 - paymentBillingPostalC...

22. 보고서에서 빈 영역을 선택한 다음 **ArcGIS Maps for Power BI** 시각화를 선택하십시오. 이 시각화 사용에 대한 조건을 수락하라는 메시지가 표시되면 지금 하십시오. **percent_suspicious** 테이블에서 **ipCountryCode** 필드를 위치(Location)로 드래그 한 다음 **SuspiciousTransactionCount** 를 색상 아래로 드래그 합니다.

VISUALIZATIONS > FIELDS

Search

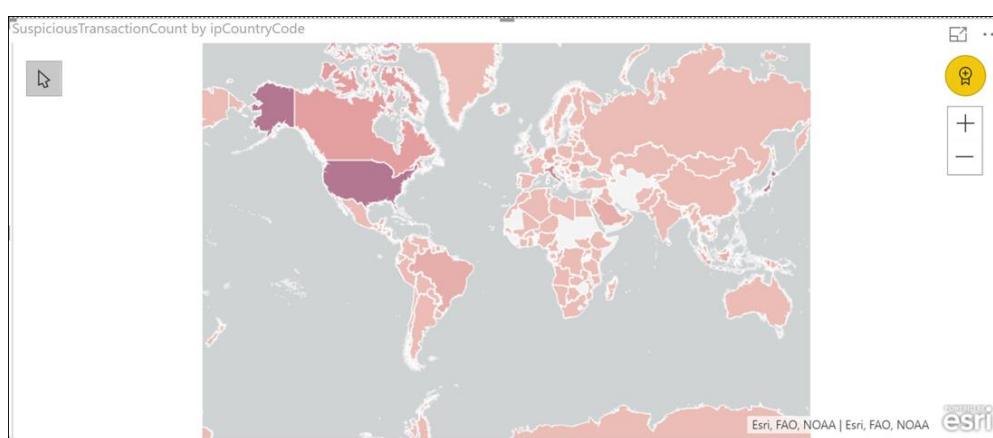
percent_suspicious

- ipCountryCode
- PercentSuspicious
- SuspiciousTransaction...
- TotalTransactionCount

scored_transactions

- accountAge
- accountCountry
- accountID
- accountPostalCode
- accountState
- browserLanguage
- cardType
- cvvVerifyResult
- digitalItemCount
- ipCountryCode
- ipPostcode
- ipState
- isProxyIP

23. ArcGIS Map 은 다음과 유사하게 표시되며 금액이 적은 국가보다 어두운 색상으로 의심스러운 거래가 많은 국가를 표시합니다.

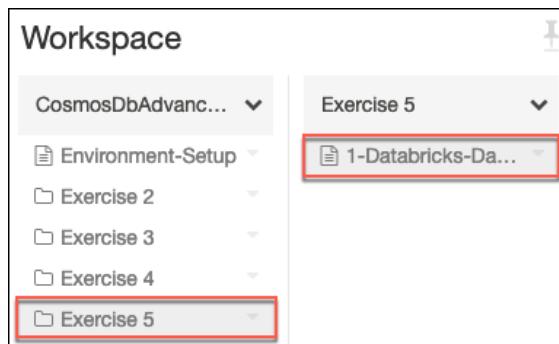


차트를 로컬 디스크에 저장할 수 있습니다. 저장된 후에는 차트를 Power BI 웹 사이트에 업로드하여 모든 차트 및 데이터 연결을 그대로 유지하면서 온라인으로 사용할 수 있습니다.

작업 2 : Azure Databricks 에서 대시 보드 만들기

이 작업에서는 Azure Databricks 노트북을 사용하여 Azure Databricks에 시각화를 표시하기 위한 대시 보드를 작성합니다.

1. 당신의 Databricks 작업 공간에서, 선택 **작업**을 한 후, 왼쪽 메뉴에서 선택 **사용자 및 사용자 계정**을.
2. 사용자 작업 공간에서 **CosmosDbAdvancedAnalytics** 폴더를 선택한 다음 **Exercise 5** 폴더를 선택하고 **1-Databricks-Dashboards** 노트북을 선택하십시오 .



3. **1-Databricks - Dashboard** 노트북에서, 이 작업의 나머지 단계를 완료하기 위한 지시 사항을 따르십시오.

실습 후

[소요 시간 : 10 분]

이 연습에서는 랩을 지원하기 위해 만든 모든 Azure 리소스를 삭제합니다. 실습 랩에 참석 한 후 제공되는 모든 단계를 수행하여 랩 리소스에 대한 계정 요금이 계속 청구되지 않도록해야합니다.

작업 1 : 리소스 그룹 삭제

1. [애저 포털](#) 왼쪽 메뉴에서 자원 그룹을 선택하여 사용했던 자원 그룹으로 이동합니다.
2. 연구 그룹의 이름을 검색하고 목록에서 선택하십시오.
3. 명령 모음에서 삭제를 선택하고 자원 그룹 이름을 다시 입력하고 삭제를 선택하여 삭제를 확인하십시오.

참조

이 내용은 아래에 게시 되어있습니다 :

<https://github.com/azure-datasolution/CloudScaleAnalytics>