datarootsio / terraform-module-azure-datalake

Terraform module for an Azure Data Lake

#terraform #azure #data-lake

213 commits | 1 branch | 0 packages | 117 releases | 2 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

sdebruyn give Databricks access to Key Vault — Latest commit cc36e3c 3 days ago

.github	fix branch for terratests	4 days ago
assets	Add docs about Power BI (#70)	last month
files	rename "transformed" to "curated"	6 days ago
test	move to helpers file	2 days ago
.gitignore	Run Terratests in CI (#76)	21 days ago
CONFIGURATION.md	allow databricks cluster config via vars	3 days ago
LICENSE.md	add MIT license (#46)	last month
Makefile	remove gotestsum from makefile	20 days ago
POWERBI.md	Add docs about Power BI (#70)	last month
README.md	rename "transformed" to "curated"	6 days ago
cosmosdb.tf	add cosmosdb container for metadata	4 days ago
data_factory.tf	Workspace URL seems to be the host instead of URL	18 days ago
databricks.tf	Only create Databricks temp storage when Synapse is provisioned	3 days ago
go.mod	move to helpers file	2 days ago
go.sum	go mod	2 days ago
key_vault.tf	give Databricks access to Key Vault	2 days ago
locals.tf	allow databricks cluster config via vars	3 days ago
main.tf	use databricks workspace url per instance	18 days ago
outputs.tf	output data factory name and ID	6 days ago
sample_data.tf	Only create Databricks temp storage when Synapse is provisioned	3 days ago

storage.tf	Only create Databricks temp storage when Synapse is provisioned	3 days ago
synapse_analytics.tf	Merge branch 'master' into feature/config-synapse	20 days ago
terraform.tfvars	Set smaller Databricks cluster in defaults	3 days ago
variables.tf	allow databricks cluster config via vars	3 days ago

README.md

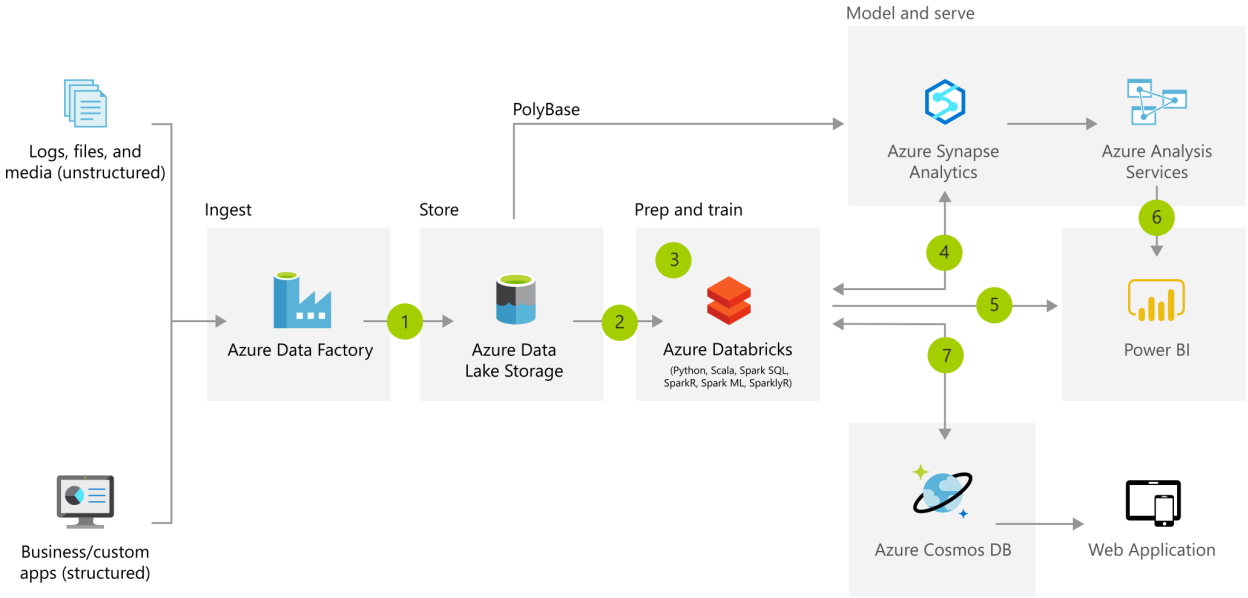
Terraform module Azure Data Lake

This is a module for Terraform that deploys a complete and opinionated data lake network on Microsoft Azure.

Components

- Azure Data Factory for data ingestion from various sources
- 3 or more Azure Data Lake Storage gen2 containers to store raw, clean and curated data
- Azure Databricks to clean and transform the data
- Azure Synapse Analytics to store presentation data
- Azure CosmosDB to store metadata
- Credentials and access management configured ready to go
- Sample data pipeline (optional)

This design is based on one of Microsoft's architecture patterns for an advanced analytics solution.



It includes some additional changes that dataroots is recommending.

- Multiple storage containers to store every version of the data (raw, cleansed, curated)
- Cosmos DB is used to store the metadata of the data as a Data Catalog
- Azure Analysis Services is not used for now as some services might be replaced when Azure Synapse Analytics Workspace becomes GA

Usage

```
module "azuredatalake" {
  source  = "datarootsio/azure-datalake/module"
  version = "~> 0.1"

  data_lake_name = "example name"
  region         = "eastus2"

  storage_replication          = "ZRS"
  service_principal_end_date   = "2030-01-01T00:00:00Z"
  databricks_cluster_node_type = "Standard_DS3_v2"
  databricks_cluster_version   = "6.5.x-scala2.11"
  databricks_token_lifetime    = 315360000
  databricks_sku               = "standard"
  data_warehouse_dtu           = "DW100c"
  cosmosdb_consistency_level   = "Session"
  cosmosdb_db_throughput       = 400
  sql_server_admin_username    = "theboss"
  sql_server_admin_password    = "ThisIsA$ecret1"
}
```

Requirements

Supported environments

This module works on macOS and Linux.

Databricks provider installation

The module is using the Databricks Terraform provider. This provider is not in the registry yet and would have to be installed manually. This can be done with the command below:

```
curl https://raw.githubusercontent.com/databrickslabs/databricks-terraform/master
```

Azure provider configuration

The following providers have to be configured:

- AzureRM
- AzureAD

You can either log in through the Azure CLI, or set environment variables as documented in the links above.

Azure CLI

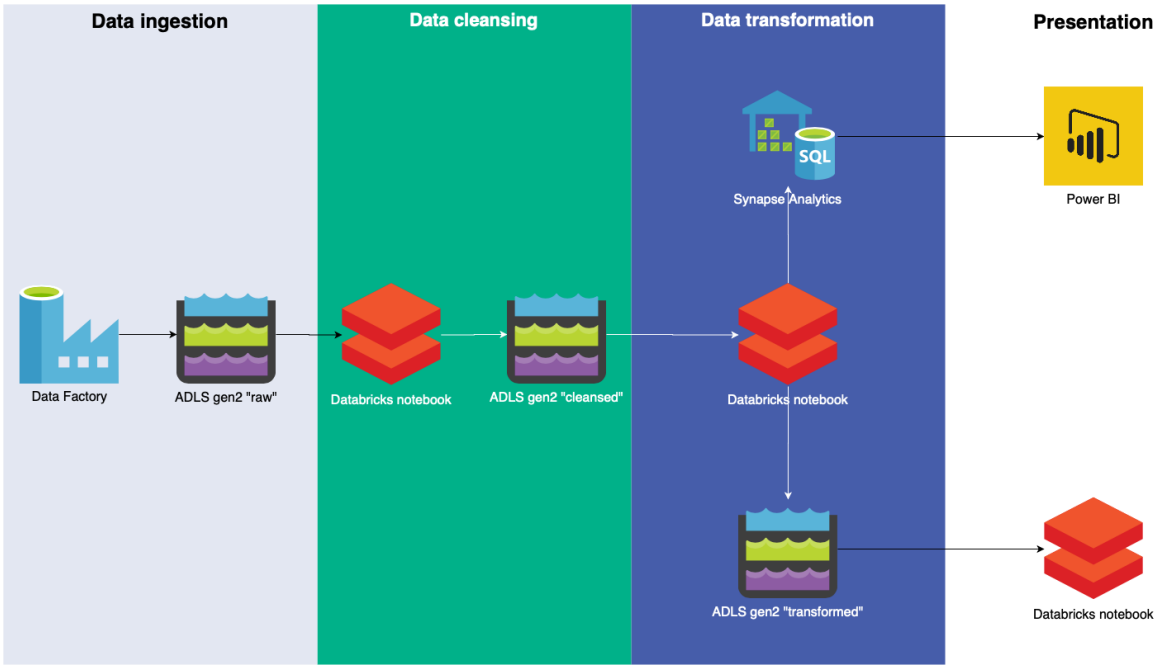
The module uses some workarounds for features that are not yet available in the Azure providers. Therefore, you need to be logged in to the Azure CLI as well. You can use both a user account, as well as service principal authentication.

PowerShell

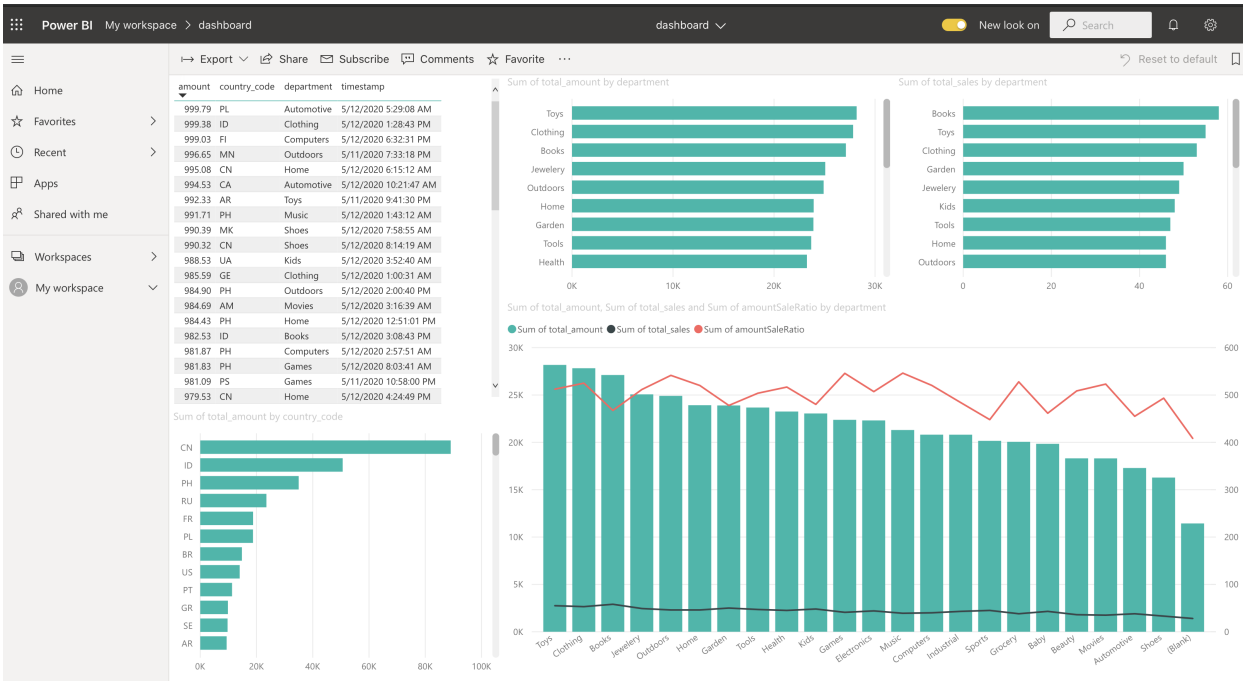
The module uses some workarounds for features that are not yet available in the Azure providers. Therefore, you need to have PowerShell installed.

Sample pipeline

The sample pipeline uses generated sales data. In the cleansing phase, personal information is removed and missing values are dealt with. In the transformation phase some aggregated values are calculated to show how each department and country is performing.



Finally, the data is presented in a Power BI dashboard. The dashboard cannot be deployed through Terraform, but you can find it in the assets folder. You can follow this guide on how to open the report and connect it to the data lake.



Configuration

The Azure tenant and subscription can be configured through the providers mentioned above. Please see Configuration for all configuration options.

Contributing

Contributions to this repository are very welcome! Found a bug or do you have a suggestion? Please open an issue. Do you know how to fix it? Pull requests are welcome as well! To get you started faster, a Makefile is provided.

Make sure to install Terraform, Azure CLI, Go (for automated testing) and Make (optional, if you want to use the Makefile) on your computer. Install tflint to be able to run the linting.

- Setup tools & dependencies: `make tools`
- Format your code: `make fmt`
- Linting: `make lint`
- Run tests: `make test` (or `go test -timeout 2h ./...` without Make)

To run the automated tests, the environment variable `ARM_SUBSCRIPTION_ID` has to be set to your Azure subscription ID.

License

MIT license. Please see LICENSE for details.