

Project Topic Team 12 - P4: Work with the database

Team Members:

- Shuxin Li - NUID 002191657
- Ruoxin Wang - NUID 002112972
- Mengjia Xu - NUID 001549384
- Boxuan Chang - NUID 001560909

Topic: Netflix Database Management System

Data Model: Relational + Document + Graph

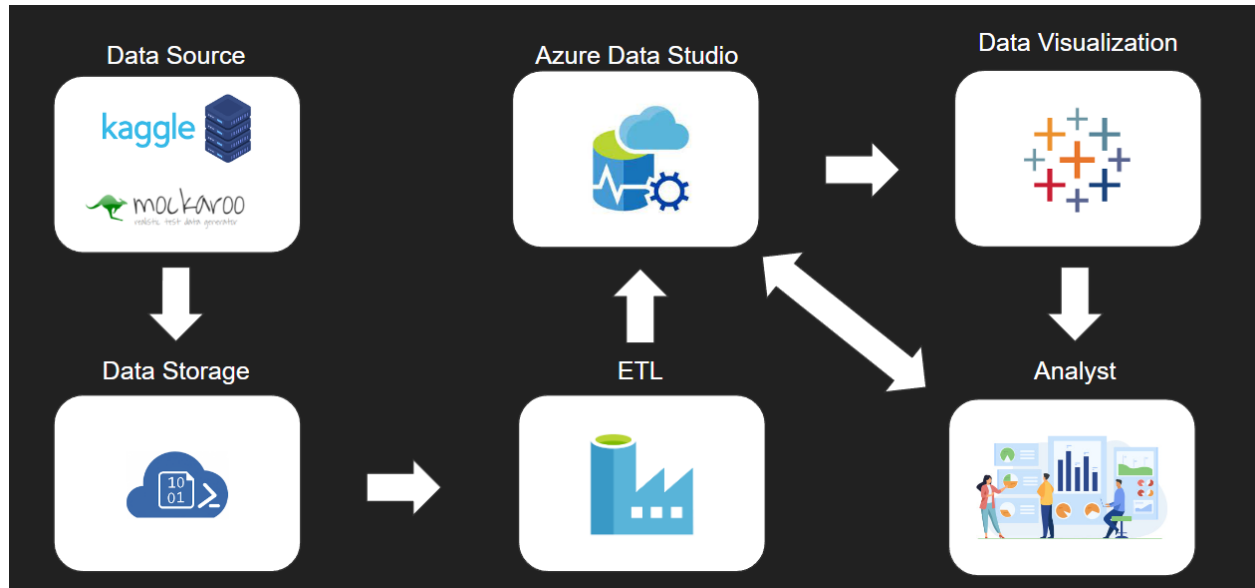
Target Platform: Azure SQL Database

Objective/Scope:

- Manage employee and group information, and implement DMLcommand(Insert, Update, Delete). Set up automated steps to synchronize group table with employee table
- Understand customer behavior, track content views, and specific customers' watching
- Filter underperforming content and track its changing trends
- Prepare monthly billing information for customers
- Evaluate and conduct trend analysis based on the plan/unit price/subscription information
- Track the top content, type, and cast of a specific time period
- Track viewing trends across different content, type, and cast
- Help Netflix make business decisions and recommendations

Visualizations Tool: Tableau/ Power BI

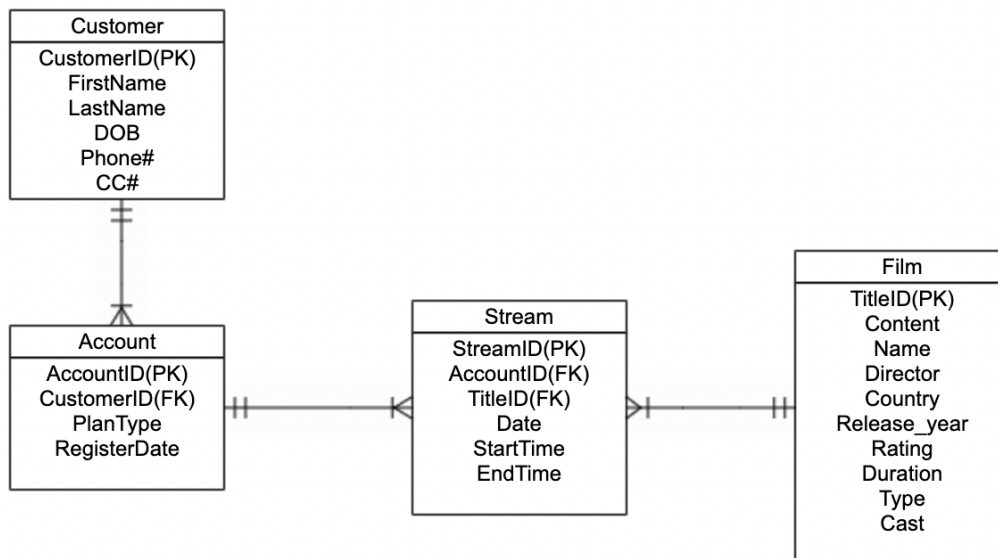
Architecture Diagram :



ERD :

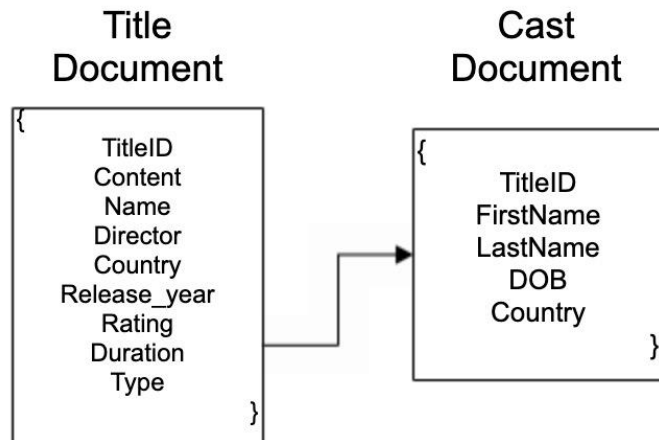
- **Relational Database**

Relational



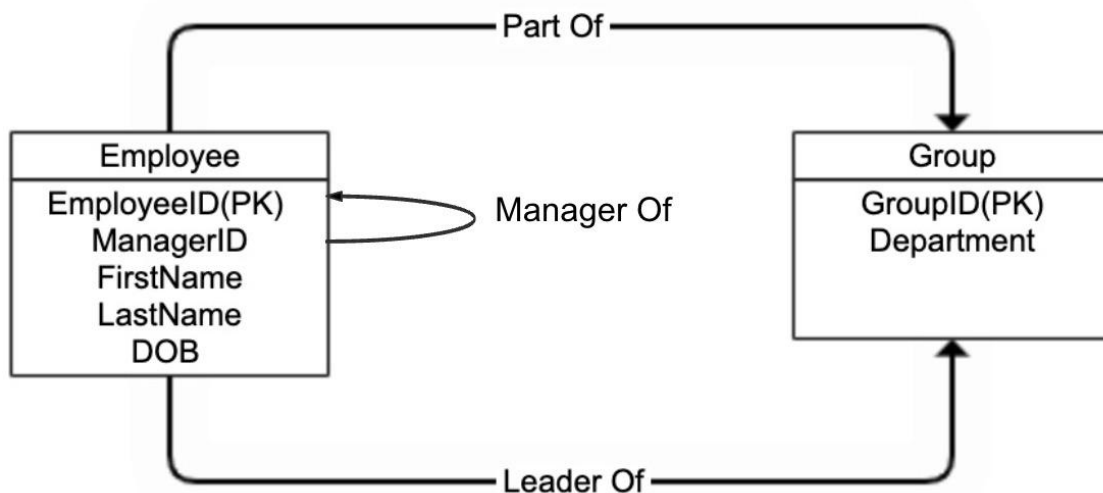
- Document Database

Document



- Graph Database

Graph



Business Rules:

Each customer can have more than one account but each account can correspond to only one customer

Each account can have none or many contents to watch

Each account can stream many tv shows or movies

Each tv show or movie can be streamed by many accounts

Each employee can have one or no manager

An employee who has no manager means he/she is the manager

Manager's ID is the same as the manager's employee ID

Employees who are in the same group will have the same manager

Each employee can be part of one or many groups

Each group can only have one manager

Entity and attribute:

Entity	Why Entity Included	How Entity Is Related To Other Entities
Customer	Store information of Customers. Using the Customer entity to maintain customer information	CustomerID is the PK of this entity. Has a one-to-many relationship with the Account entity. Each customer can have many accounts
Account	Store information of Account. Using the attribute "RegisterDate" to create an auto step to remind the customer to renew their plan	AccountID is the PK of this entity. CustomerID is the FK and references CustomerID in the Customer entity. Each account can only be owned by one customer. Has a many-to-many relationship with Title entity.
Title	Store information of each tv show or movie. Include TitleID, ContentID, TypeID, Cast. Using this entity to maintain title information.	TitleID is the PK of this entity. Has a one-to-many relationship with the Content entity and Type entity. Has ContentID, TypeID as FK and reference ContentID, TypeID in Content and Type entity. Using embedding method to connect with Cast entity.

Type	Store Type information. Such as name and limit. Using limit to check if customers' DOB is available to watch the specific tv show or movie.	TypeID is the PK of this entity. Has a one-to-many relationship with the Title entity.
Content	Store Content information.	ContentID is the PK of this entity. Has a one-to-many relationship with the Title entity.
Cast	Store Cast information.	Use the embedding method to be stored in the Title entity.
Stream	Store Stream information. Record StartTime and EndTime of each stream. Using those two attributes to analyze the viewing and changing trends for each tv show or movie.	The bridge entity between the many-to-many relationship of the Account entity and the Title entity. StreamID is the PK of this entity. AccountID and TitleID are FK and reference to the AccountID, TitleID in the Account, Title entity.
Employee	Store Employee information. Each employee has a unique EmployeeID. If the employee has a manager, his/her ManagerID will be the manager's EmployeeID.	One node of the graph database. An employee is part of a group. A manager is a leader of a group. A manager is a manager of an employee.
Group	Store group information. Each group has a unique GroupID and a ManagerID. Employees in the same group will have the same ManagerID.	One node of the graph database. A group is led by a manager. And grouped by employees

Implementation:

We used Azure data factory to build **three** data pipelines for importing/converting data from csv files to corresponding databases(relational database, document database, and graph database).

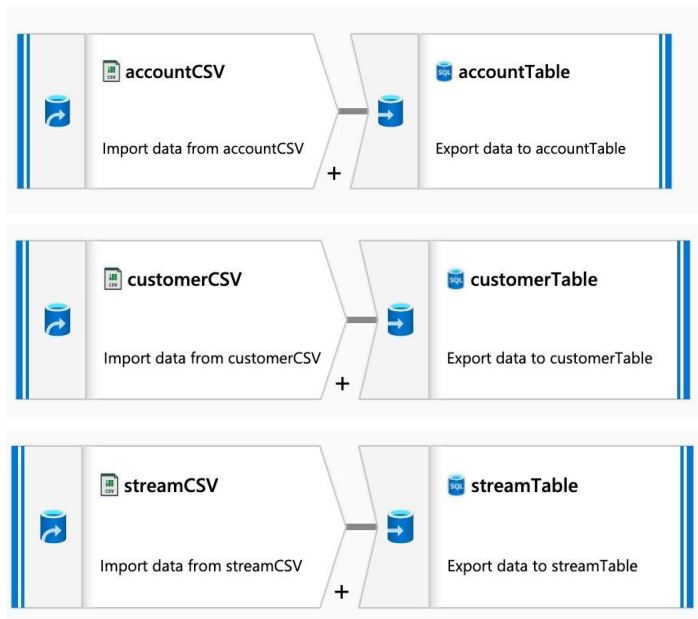
Relational Database

For importing data to the relational database, we create three data flows:

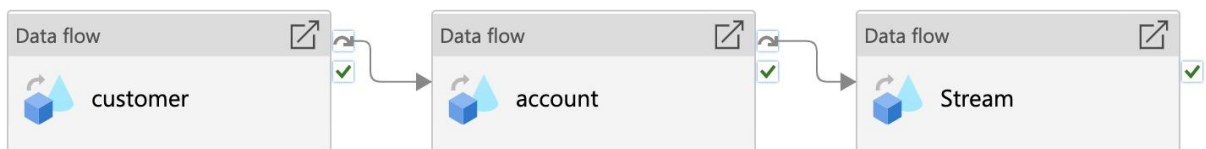
1. accountCSV to [account] table
2. CustomerCSV to [customer] table
3. streamCSV to [stream] table

And the dataflow running sequence is determined so that the data which has dependencies will be imported later than its dependencies.

Data Flows:



Pipelines:



Data Refresh/Trigger:

Edit trigger

Name *

RelationalTrigger

Description

Type *

ScheduleTrigger

Start date * ⓘ

11/17/22 23:00:00

Time zone * ⓘ

Pacific Time (US & Canada) (UTC-8)

ⓘ This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Trigger runs

All Schedule Tumbling window Storage events Custom events Refresh Edit columns

Pacific Time (US & C... : Last 24 hours Trigger name : RelationalTrigger Status : All Runs : Latest runs Export to CSV

Showing 1 - 1 items

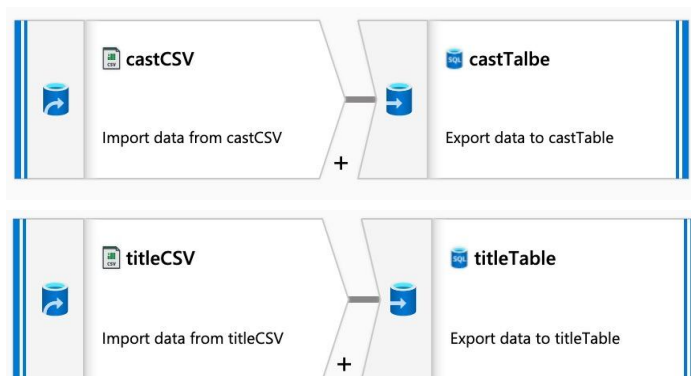
Trigger name ↑↓	Trigger type	Trigger time ↑↓	Status ↑↓	Pipelines	Run	Message
RelationalTrigger	Schedule trigger	Nov 18, 2022, 11:00:01	✓ Succeeded	1	Original	

Document Database

For importing data to the document database, we create two data flows and a stored procedures:

1. titleCSV to [title] table
2. castCSV to [cast] table
3. A stored procedure will be called to do the embedding work, then write to [film] table

Data Flows:



Pipelines:



Create Stored Procedure

```
CREATE PROCEDURE embedding
AS
DECLARE @json VARCHAR(MAX);
SET @json = (SELECT DISTINCT Title_id, Content, Name, Director, t2.Country,
    Release_year, Rating, Duration, Type,
    (SELECT DISTINCT First_name, Last_name, DOB, c.Country
    FROM [dbo].[Cast] c
    LEFT JOIN [dbo].[Title] t
    ON c.Title_id = t.Title_id
    WHERE t.Title_id = t2.Title_id
    FOR JSON PATH) AS CAST
FROM [dbo].[Title] t2
GROUP BY t2.Title_id, Content, Name, Director, t2.Country, Release_year, Rating,
Duration, Type
FOR JSON PATH);
INSERT INTO Film
SELECT *
FROM OPENJSON ( @json )
WITH (
    Title_id    VARCHAR(10)    '$.Title_id',
    Content     VARCHAR(20)     '$.Content',
    Name        VARCHAR(50)     '$.Name',
    Director    VARCHAR(50)     '$.Director',
    Country     VARCHAR(50)     '$.Country',
    Release_year INT            '$.Release_year',
    Rating      VARCHAR(10)     '$.Rating',
    Duration    VARCHAR(50)     '$.Duration',
    Type        VARCHAR(50)     '$.Type',
    [CAST]      NVARCHAR(MAX)   AS JSON
)
```


Formatted JSON

Messages

```
{
  "Title_id": "s1099",
  "Content": "Movie",
  "Name": "Mandela",
  "Director": "Madonne Ashwin",
  "Country": "India",
  "Release_year": 2021,
  "Rating": "TV-14",
  "Duration": "140 min",
  "Type": "Comedies",
  "CAST": [
    {
      "First_name": "Sheela",
      "Last_name": "Rajkumar",
      "DOB": "1992-06-14",
      "Country": "India"
    },
    {
      "First_name": "Yogi",
      "Last_name": "Babu",
      "DOB": "1985-07-22",
      "Country": "India"
    }
  ]
},
{
  "Title_id": "s1142",
  "Content": "Movie",
  "Name": "Nazir",
  "Director": "Rajou Nambiar"
```

✔ Query succeeded | 0s

Data Refresh/Trigger:

Edit trigger

Name *

DocumentTrigger

Description

Type *

ScheduleTrigger

Start date * ⓘ

11/17/22 23:00:00

Time zone * ⓘ

Pacific Time (US & Canada) (UTC-8)

ⓘ

This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Trigger runs

All

Schedule

Tumbling window

Storage events

Custom events

Refresh

Edit columns

Pacific Time (US & C... : Last 7 days

Trigger name : All

Status : All

Runs : Latest runs

×

Showing 1 - 1 items

Trigger name ↑↓	Trigger type	Trigger time ↑↓	Status ↑↓	Pipelines	Run
DocumentTrigger	Schedule trigger	Nov 17, 2022, 11:00:00	✔ Succeeded	1	Original

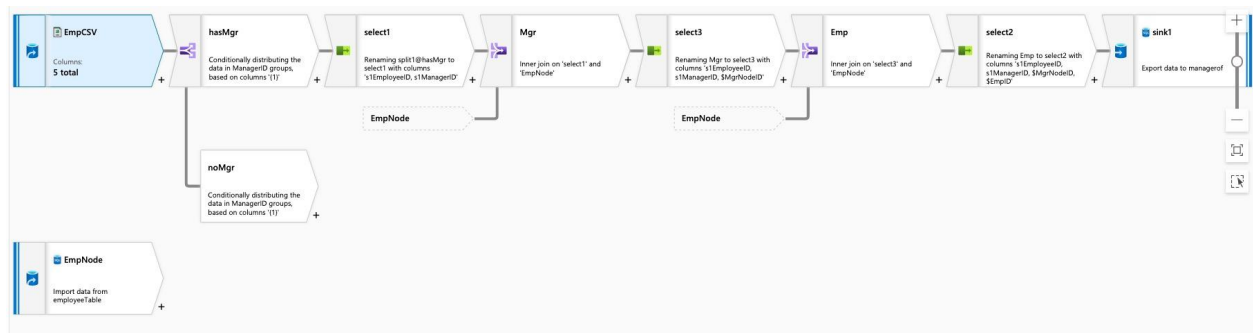
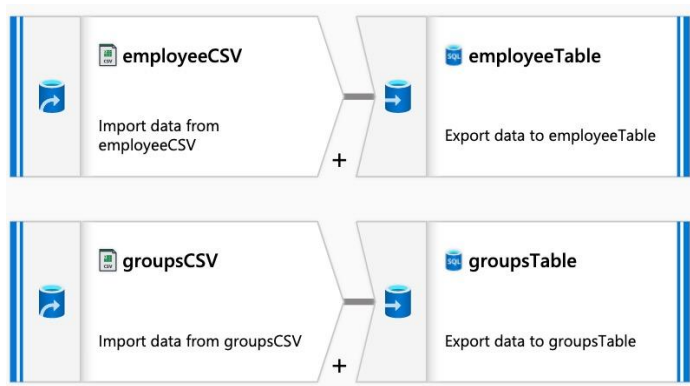
Graph Database

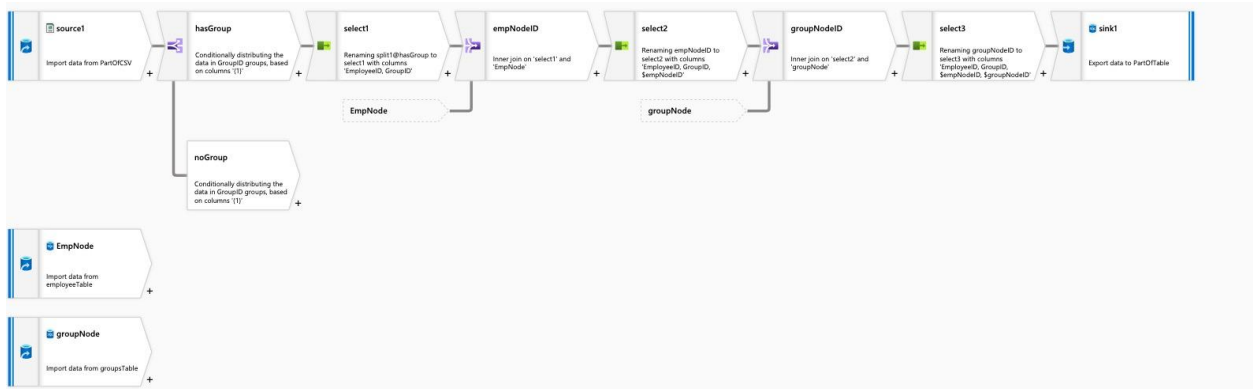
For importing data to the graph database, we create four data flows:

1. NodeCSV to node tables(employee table and group table)
2. edgeCSV to edge table
 - a. edgeCSV to edge table[managerOf]
 - b. edgeCSV to edge table[partOf]
 - c. edgeCSV to edge table[leaderOf]

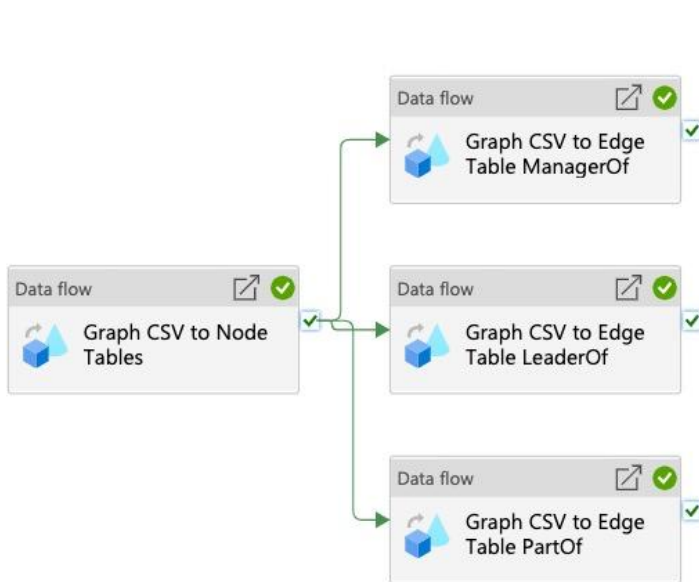
And the dataflow running sequence is determined so that the node tables get data before edge tables get data.

Data Flows:





Pipelines:



Data Refresh/Trigger:

Edit trigger

Name *

GraphTrigger

Description

Type *

ScheduleTrigger

Start date * ⓘ

11/17/22 23:00:00

Time zone * ⓘ

Pacific Time (US & Canada) (UTC-8) ▾

ⓘ

This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Trigger runs

AllScheduleTumbling windowStorage eventsCustom events

RefreshEdit columns

Pacific Time (US & C... : Last 24 hours

Trigger name : GraphTrigger

Status : All

Runs : Latest runs

Export to CSV

Showing 1 - 1 items

Trigger name ↑↓	Trigger type	Trigger time ↑↓	Status ↑↓	Pipelines	Run	Message
GraphTrigger	Schedule trigger	Nov 18, 2022, 10:59:!	✔ Succeeded	1	Original	