**DATA ENGINEER ASSESMENT**

**PARHAN**

# Data and Query Task (Basic)

1. Create script to create table for each object
   1. Employee
   2. PositionHistory

Answer:

CREATE TABLE Employee (

Id INT PRIMARY KEY,

EmployeeId VARCHAR(10) UNIQUE NOT NULL,

FullName VARCHAR(100) NOT NULL,

BirthDate DATE NOT NULL,

Address VARCHAR(500)

);

CREATE TABLE PositionHistory (

Id INT PRIMARY KEY,

FOREIGN KEY (EmployeeId) REFERENCES Employee(EmployeeId),

PosId VARCHAR(10) NOT NULL,

PosTitle VARCHAR(100) NOT NULL,

EmployeeId VARCHAR(10) NOT NULL,

StartDate DATE NOT NULL,

EndDate DATE

);

1. Create insert script to inserting data into each table (*Employee and PositionHistory*)

**Employee**

INSERT INTO Employee (Id, EmployeeId, FullName, BirthDate, Address) VALUES

(1, '10105001', 'Ali Anton', '1982-08-19', 'Jakarta Utara'),

(2, '10105002', 'Rara Siva', '1982-01-01', Jakarta Selatan'),

(3, '10105003', 'Rini Aini', '1982-02-20', 'Padang'),

(4, '10105004', 'Budi', '1982-02-22', 'Jakarta Barat);

**PositionHistory**

INSERT INTO PositionHistory (Id, PosId, PosTitle, EmployeeId, StartDate, EndDate) VALUES

(1, '50000', 'IT Manager', '10105001', '2022-01-01', '2022-02-28'),

(2, '50001', 'IT Sr. Manager', '10105001', '2022-03-01', '2022-12-31'),

(3, '50002', 'Programmer Analyst', '10105002', '2022-01-01', '2022-02-28'),

(4, '50003', 'Sr. Programmer Analyst', '10105002', '2022-03-01', '2022-12-31'),

(5, '50004', 'IT Admin', '10105003', '2022-01-01', '2022-02-28'),

(6, '50005', 'IT Secretary', '10105003', '2022-03-01', '2022-12-31');

1. Create query to display all employee (*EmployeeId, FullName, BirthDate, Address*) data with their

**current** position information (*PosId, PosTitle, EmployeeId, StartDate, EndDate*).

SELECT

e.EmployeeId,

e.FullName,

e.BirthDate,

e.Address,

p.PosId,

p.PosTitle,

p.EmployeeId,

p.StartDate,

p.EndDate

FROM

Employee e

JOIN

PositionHistory p ON e.EmployeeId = p.EmployeeId

WHERE

(p.EmployeeId, p.EndDate) IN (

SELECT

EmployeeId,

MAX(EndDate)

FROM

PositionHistory

GROUP BY

EmployeeId

)

ORDER BY

e.EmployeeId;

## Tables and Samples Data

1. Employee Table

**Employee Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Name** | **Type** | **Length** | **Mandatory** | **Identity** | **Key** | **Sample Data** |
| 1 | Id | Integer |  | Y | Y | Primary Key | 1 |
| 2 | EmployeeId | String | 10 | Y | N | Unique Key | 10105001 |
| 3 | FullName | String | 100 | Y | N |  | Ali Anton |
| 4 | BirthDate | Date |  | Y | N |  | 19-Aug-82 |
| 5 | Address | String | 500 | N | N |  | Jakarta Utara |

**Employee Sample Data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **EmployeeId** | **FullName** | **BirthDate** | **Address** |
| 1 | 10105001 | Ali Anton | 19-Aug-82 | Jakarta Utara |
| 2 | 10105002 | Rara Siva | 1-Jan-82 | Mandalika |
| 3 | 10105003 | Rini Aini | 20-Feb-82 | Sumbawa Besar |
| 3 | 10105004 | Budi | 22-Feb-82 | Mataram Kota |

1. Position Table

**Position History Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Name** | **Type** | **Length** | **Mandatory** | **Identity** | **Key** | **Sample Data** |
| 1 | Id | Integer |  | Y | Y | Primary Key | 1 |
| 2 | PosId | String | 10 | Y | N |  | 50001 |
| 3 | PosTitle | String | 100 | Y | N |  | IT Sr. Manager |
| 4 | EmployeeId | String | 10 | Y | N |  | 10105001 |
| 5 | StartDate | Date |  | Y | N |  | 1-Mar-2022 |
| 6 | EndDate | Date |  | Y | N |  | 31-Dec-2022 |

**Position History Sample Data**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **I**  **d** | **PosId** | **PosTitle** | **EmployeeID** | **StartDate** | **EndDate** |
| 1 | 50000 | IT Manager | 10105001 | 1-Jan-2022 | 28-Feb-2022 |
| 2 | 50001 | IT Sr. Manager | 10105001 | 1-Mar-2022 | 31-Dec-2022 |
| 3 | 50002 | Programmer Analyst | 10105002 | 1-Jan-2022 | 28-Feb-2022 |
| 4 | 50003 | Sr. Programmer Analyst | 10105002 | 1-Mar-2022 | 31-Dec-2022 |
| 5 | 50004 | IT Admin | 10105003 | 1-Jan-2022 | 28-Feb-2022 |
| 6 | 50005 | IT Secretary | 10105003 | 1-Mar-2022 | 31-Dec-2022 |

# ETL, Data Warehouse and Analytics Task

Build and design simple Data Warehouse and analytics data, data source came from 2 different source

1. Employee data, source from Azure
2. Training history, source from GCP (Google Cloud Platform)

## Requirement and Details:

1. Azure Employee data is using SQL Server as Database Server. This is database for saving employment data like their employee number, name, birthdate, position/job etc. Each employee has one record for their employment data.
2. Training History data is using Google Worksheet as data platform. The worksheet contains historical data about training that completed by employee. Each employee may have more than one records for their training history data.
3. Design simple ETL Flow that compile these two data sources into Data Warehouse
4. Design simple report that displays historical training data
5. Design simple dashboard that displays
   1. Total employee completed training each month
   2. Total training each month

Answer:

1. **Employee Data (Azure)**

* Source: SQL Server Database
* Description: This database stores employee data such as employee number, name, birthdate, position/job, etc. Each employee has one record for their employment data.

1. **Training History (GCP)**

* Source: Google Worksheet
* Description: The worksheet contains historical data about training that is completed by employees. Each employee may have more than one record for their training history data.

**Steps to Build and Design the Data Warehouse and Analytics System**

**1. Extract Data from Sources**

a. Extract Employee Data from SQL Server

* SQL Query to Extract Employee Data:

SELECT EmployeeID, Name, Birthdate, Position

FROM Employee;

b. Extract Training History Data from Google Worksheet

* Using Google Sheets API:

Use a library like gspread in Python to access the Google Worksheet, Example:

import gspread

from oauth2client.service\_account import ServiceAccountCredentials

# Use credentials to create a client to interact with the Google Drive API

scope = ["https://spreadsheets.google.com/feeds", 'https://www.googleapis.com/auth/drive']

creds = ServiceAccountCredentials.from\_json\_keyfile\_name('path\_to\_credentials.json', scope)

client = gspread.authorize(creds)

# Find a workbook by name and open the first sheet

sheet = client.open("TrainingHistory").sheet1

# Extract all records

training\_data = sheet.get\_all\_records()

**2. Transform Data for Consistency**

a. Data Normalization

* Ensure date formats are consistent (e.g., YYYY-MM-DD).
* Normalize the case of strings (e.g., making all text fields uppercase/lowercase).
* Validate and clean data to remove any inconsistencies or errors.

b. Example Transformation Code

import pandas as pd

# Example data frames

employee\_df = pd.DataFrame({

'EmployeeID': [10105001, 10105002, 10105003],

'Name': ['Alice', 'Bob', 'Charlie'],

'Birthdate': ['1985-01-01', '1990-02-02', '1988-03-03'],

'Position': ['IT Manager', 'Programmer Analyst', 'IT Admin']

})

training\_df = pd.DataFrame(training\_data)

# Convert dates to datetime objects

employee\_df['Birthdate'] = pd.to\_datetime(employee\_df['Birthdate'])

training\_df['CompletionDate'] = pd.to\_datetime(training\_df['CompletionDate'])

# Normalize string fields

employee\_df['Name'] = employee\_df['Name'].str.upper()

employee\_df['Position'] = employee\_df['Position'].str.upper()

training\_df['TrainingName'] = training\_df['TrainingName'].str.upper()

**3. Load Data into Data Warehouse**

a. Create Data Warehouse Schema

Design the schema for the Data Warehouse.Example schema in SQL:

CREATE TABLE Employee (

EmployeeID INT PRIMARY KEY,

Name VARCHAR(100),

Birthdate DATE,

Position VARCHAR(100)

);

CREATE TABLE TrainingHistory (

TrainingID INT PRIMARY KEY AUTO\_INCREMENT,

EmployeeID INT,

TrainingName VARCHAR(100),

CompletionDate DATE,

FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)

);

**b. Load Data into Data Warehouse**

* Use an ETL tool or script to load the transformed data into the Data Warehouse. Example using Python with SQLAlchemy:

from sqlalchemy import create\_engine

# Create an engine that stores data in the local directory's sqlite database

engine = create\_engine('sqlite:///data\_warehouse.db')

# Load data into the data warehouse

employee\_df.to\_sql('Employee', con=engine, if\_exists='replace', index=False)

training\_df.to\_sql('TrainingHistory', con=engine, if\_exists='replace', index=False)

**4. Design Simple Report for Historical Training Data**

* SQL Query to Generate Report:

SELECT e.EmployeeID, e.Name, t.TrainingName, t.CompletionDate

FROM Employee e

JOIN TrainingHistory t ON e.EmployeeID = t.EmployeeID

ORDER BY e.EmployeeID, t.CompletionDate;

**5. Design Simple Dashboard**

**a. Total Employees Completed Training Each Month And Year**

* SQL Query:

SELECT YEAR(CompletionDate) AS Year, MONTH(CompletionDate) AS Month, COUNT(DISTINCT EmployeeID) AS TotalEmployees

FROM TrainingHistory

GROUP BY YEAR(CompletionDate), MONTH(CompletionDate)

ORDER BY Year, Month;

A graph of different colored lines

Description automatically generated with medium confidence

**b. Total Training Each Month And Year**

* SQL Query:

SELECT YEAR(CompletionDate) AS Year, MONTH(CompletionDate) AS Month, COUNT(TrainingName) AS TotalTrainings

FROM TrainingHistory

GROUP BY YEAR(CompletionDate), MONTH(CompletionDate)

ORDER BY Year, Month;

A graph with lines and numbers

Description automatically generated with medium confidence