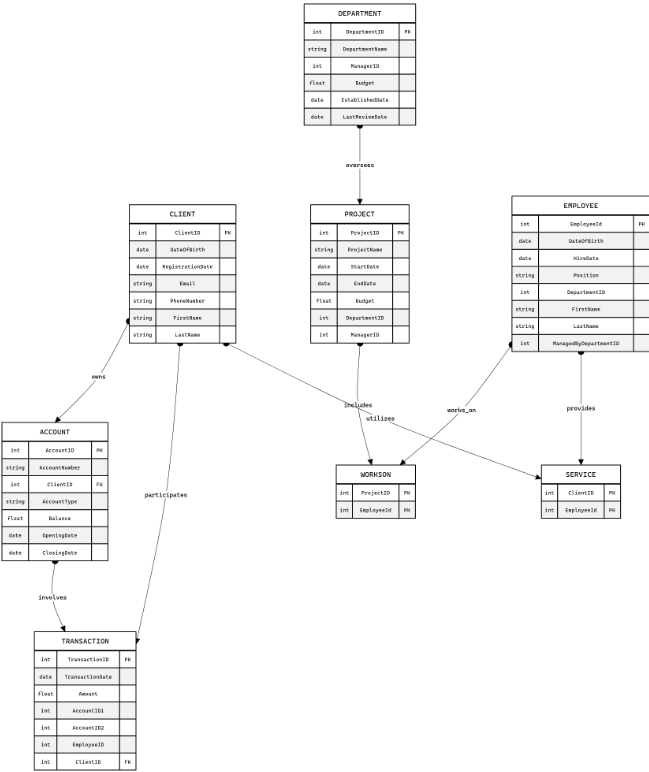
דוח שלב 4 :

תרשימים – אליסף

A diagram of a company

Description automatically generatedרצינו לשלב בין 2 הפרויקטים

וראינו שיש סהכ שתי טבלאות דומות שנרצה באמצעותם לבצע את האינטגרציה.

Clients/Customers

Employees/Workers

ע"מ לעשות את האינטגרציה החלטנו על השדות החדשים של הטבלה, השארנו את שמות הטבלאות שלנו כפי שהיו



בתמונה המצורפת ניתן לראות את השדות של הטבלה המאוחדת: customer

שדה שנוסף לדוגמא: Email

שדה מהטבלה שלנו שנשאר לדוגמא: customerName

שדה מהטבלה שלהם שתירגמנו לטבלה שלנו: firstName, lastName שונה להיות customerName

הטבלה השנייה ששונתה: employee שאוחדה עם workers

בתמונה המצורפת ניתן לראות את השדות של הטבלה המאוחדת: workers

שדה שנוסף לדוגמא: hireDate

שדה מהטבלה שלנו שנשאר לדוגמא: workerName

שדה מהטבלה שלהם שתירגמנו לטבלה שלנו: firstName, lastName שונה להיות workerName

**הסבר קובץ integrate:**

הצגת נתונים:

* הקובץ מתחיל עם פקודות SELECT \* שמציגות את כל הרשומות מהטבלאות השונות כגון account, client, customers, department, employee, workers, reporters, transaction, ו-service. זה נעשה כדי לבדוק את המידע הקיים בטבלאות לפני ביצוע שינויים.
* הוספת עמודות לטבלה customers:
  + הטבלה customers עוברת שינוי בעזרת הפקודה ALTER TABLE, שמוסיפה לה עמודות חדשות: DateOfBirth (תאריך לידה), Email (אימייל), ו-PhoneNumber (מספר טלפון).
  + בנוסף, הפקודה MODIFY משנה את סוג הנתונים של העמודה customerName ל-VARCHAR2(40).
* הכנסת נתונים לטבלה CUSTOMERS:
  + פקודת ה-INSERT INTO משמשת להוספת רשומות לטבלה CUSTOMERS. הנתונים שמוכנסים לטבלה מתקבלים משאילתת SELECT המשלבת נתונים מהטבלה CLIENT והטבלה ACCOUNT. המטרה היא לאחד את נתוני הלקוח והחשבון לטבלה אחת.

**הוספת עמודות ושינוי מבנה בטבלה workers**:

* הטבלה workers עוברת שינויים בעזרת פקודת ALTER TABLE שמוסיפה עמודות כמו DateOfBirth, HireDate, DepartmentID, ו-ManagedByDepartmentID.
* כמו כן, הפקודות MODIFY משנות את סוג הנתונים של העמודות WorkerRole ו-WorkerName.

**הכנסת נתונים לטבלה workers**:

* הנתונים שמוכנסים לטבלה workers מתקבלים משאילתת SELECT מהטבלה EMPLOYEE. הנתונים המועתקים כוללים מידע על תאריכים, מחלקות, ומידע נוסף על העובדים.

**הוספת מפתח זר**:

* הטבלה ACCOUNT מקבלת מפתח זר (FOREIGN KEY) בשם fk\_account\_customerid, שמקשר את העמודה ClientID לטבלה CUSTOMERS.

**הסרת מפתח זר בטבלה TRANSACTION**:

* הפקודה ALTER TABLE מסירה את המפתח הזר הקיים בטבלה TRANSACTION, ופקודת DELETE מוחקת רשומות מסוימות מהטבלה TRANSACTION בהתבסס על תנאים שנלקחים משאילתת משנה.
* בסוף קובץ האינטגרציה (intgrate.sql) מחקנו את 2 הטבלאות client ו- employee

מבט 1:

CREATE VIEW CustomerAccountsView AS  
SELECT  
c.CustomerID,  
c.customername,  
c.Email,  
c.PhoneNumber,  
a.AccountID,  
a.AccountNumber,  
a.AccountType,  
a.Balance,  
a.OpeningDate,  
a.ClosingDate  
FROM  
Customers c  
JOIN  
Account a ON c.CustomerID = a.ClientID;

View שנוצר בשם CustomerAccountsView מחבר בין נתוני לקוחות (Customers) וחשבונות (Account). התצוגה כוללת פרטים כמו מזהה לקוח, שם, מייל, מספר טלפון, מזהה חשבון, מספר חשבון, סוג חשבון, יתרה ותאריכי פתיחה וסגירה של החשבון.

שאילתא 1:

SELECT c.CustomerID, c.customername, COUNT(a.AccountID) AS ActiveAccountsCount, SUM(a.Balance) AS TotalBalance

FROM CustomerAccountsView c JOIN Account a ON c.CustomerID = a.clientid WHERE a.Balance > 10000 AND a.ClosingDate IS NULL

GROUP BY c.CustomerID, c.customername ORDER BY TotalBalance DESC;

השאילתא באה לחשב את מספר החשבונות הפעילים ואת הסכום הכולל של היתרה עבור כל לקוח.

SELECT c.CustomerID, c.customername, COUNT(DISTINCT a.AccountType) AS DifferentAccountTypesCount, MIN(a.OpeningDate) AS FirstAccountOpeningDate

FROM CustomerAccountsView c JOIN Account a ON c.CustomerID = a.clientid

WHERE a.OpeningDate > DATE '2020-01-01' GROUP BY c.CustomerID, c.customername HAVING COUNT(DISTINCT a.AccountType) > 1 ORDER BY FirstAccountOpeningDate ASC;

השאילתא באה לחשב את מספר סוגי החשבונות השונים של כל לקוח ולמצוא את תאריך פתיחת החשבון הראשון שלו.

מבט שני:  
מאפשר גישה משולבת למידע על לקוחות, סניפים, הלוואות, ועובדים, ומסייעת בניתוח מקיף של הקשרים בין הלקוחות לשאר הנתונים

CREATE VIEW OriginalDepartmentView AS

SELECT

c.customerID,

c.customerName,

c.accountID,

c.joinDate,

c.brancheID,

b.brancheLocation,

b.openedDate AS brancheOpenedDate,

l.loanID,

l.loanAmount,

l.loanDate,

w.workerID,

w.workerName,

w.workerRole,

w.workerAge,

w.departmentID,

w.hireDate

FROM

customers c

JOIN

branche b ON c.brancheID = b.brancheID

LEFT JOIN

loans l ON c.customerID = l.customerID

LEFT JOIN

worker w ON w.brancheID = c.brancheID

שאילתא 1: השאילתה מחשבת את מספר הלקוחות בכל סניף:

SELECT

b.brancheLocation,

COUNT(c.customerID) AS numberOfCustomers

FROM

OriginalDepartmentView c

JOIN

branche b ON c.brancheID = b.brancheID

GROUP BY

b.brancheLocation;

שאילתא 2 :מציאת הלוואות של לקוחות בסניף מסוים(תל אביב):

SELECT

customerName,

loanAmount,

loanDate

FROM

OriginalDepartmentView

WHERE

BrancheLocation = 'tel aviv'