Shengjie Quan, Alex Tanchevski, and Kelly Doan

Dr. Krishnasamy

CSE 3241

Project Phase 1: Database Design

**Description:**

A grocery store company has many stores, and they are uniquely identified by store ID. Every store has a specific location, which further consists of street, city, state, and zip code. The store phone number is also tracked. Customers can become members by signing up for membership at one of these stores, and a single store can start memberships for multiple customers. Every member must start membership at one of these stores, but not every store has to start membership for customers. Members are identified by member ID, and their first name, middle initial, last name, phone number, address (street, city, state, and zip code), and birthday are kept as well.

Each store has many employees, and employees can only work at one store maximum. Every store must have employees. However, some employees don’t work at a store but rather in a different setting (such as the headquarters or offices). Employees are identified by their employee ID, and their first name, middle initial, last name, phone number, and birthday are kept. The total daily number of hours the employee has worked can also be derived. Dependents of employees are also tracked, where every dependent is connected to an employee but not every employee has a dependent. An employee can have more than one dependent, but a dependent is only linked to one employee. Dependents are identified by their name in addition to the employee ID that they are dependents of. Their sex, birthday, and relationship to the employee are also recorded. The company also has multiple departments. Every department has employees and can have multiple, whereas an employee works for only one department. However, not every employee works for a department (such as some of the employees who work in stores). Departments are identified by department ID, and their name and location (street, city, state, and zip code) are tracked as well. Furthermore, store employees work shifts and can have more than one. However, not every employee works shifts since only store employees work shifts. Every shift needs at least one employee, but the same shift can have multiple employees. The number of hours that an employee works a shift is tracked. Shifts are identified by a shift ID, and their start time, end time, and the hourly payment rate for that particular shift are kept. Shifts also belong to a particular store. Every shift must belong to a store, and every store has shifts. A shift only belongs to one store, while stores can have multiple shifts.

The company has multiple suppliers as well, which are uniquely identified by supplier ID, and their name is also kept in the database. Each supplier has multiple warehouses, but every warehouse belongs to one supplier. These warehouses are identified by a warehouse ID, and the database also keeps their location (street, city, state, and zip code). First, the stores place orders with the suppliers. Every store places orders and can place multiple, while every order must have been placed by a store but only belongs to one store. Each of these orders must be directed towards one specific supplier, and every supplier receives orders. Suppliers can receive multiple orders as well. Each order is identified by an order ID, and the date that the order is placed is recorded. Each order also consists of items, but not every item is in an order. Orders can consist of multiple items while items can be included in multiple orders.

Next, trucks take shipments from the warehouses, and the shipments are delivered to the stores. Trucks are uniquely identified by a license plate number, but their capacity and type are also recorded. A truck can take multiple shipments, but a particular shipment would only be carried by one truck. Also, not every truck takes shipments, but every shipment must be carried by a truck. Shipments are identified by a shipment ID, and the database also keeps the date that the shipment is made and can derive the total weight of the shipment. Every shipment departs from a warehouse, but not every warehouse sends out shipments. Also, a shipment only departs from one warehouse, but a warehouse can send out multiple shipments. Furthermore, every shipment is delivered to one specific store, and every store receives shipments. Stores can receive multiple shipments as well. Every shipment also contains specific items, but not every item belongs to a shipment. Shipments can include multiple items, and items can be in multiple shipments. Each of these items is tracked and identified with an item ID, and the name and weight of the item is also recorded.

These items are sold by suppliers. Every item must be sold by a supplier, and every supplier sells items. Also, a supplier can sell multiple items, but particular items are sold by one supplier. Each store has particular items in stock, and the quantity of each item in stock at one store is recorded. Every store has items, but not every item is found at a store. Moreover, stores can have multiple items, and items can be found at multiple stores. In addition, prescriptions are filled at stores, and they are identified by a prescription ID, while the prescribing doctor and patient name are also recorded. Every prescription must be filled at a store, but not every store fills prescriptions. Also, a store can fill multiple prescriptions, but one particular prescription is only filled at one store.

Stores also make their own weekly ads to be sent to the members. Every store makes a weekly ad and can have more than one, whereas each weekly ad belongs to only one store. Every weekly ad is then sent to members, but not every member receives a weekly ad. Weekly ads can be sent to multiple members, and members can receive multiple weekly ads. The weekly ad is identified by a weekly ad ID. A weekly ad includes coupons for specific items as well. Every weekly ad includes coupons, but not every coupon is associated with a weekly ad. Weekly ads can also include multiple coupons, and coupons can be associated with multiple weekly ads. These coupons have a coupon ID, and their expiration date and specific discount are tracked. Every coupon is for a particular item, but not every item has a coupon. Also, coupons are for one item, but items can have multiple coupons.

**ER Diagram:** on separate PDF file

**Additional Constraints:** also listed on ER diagram

1. A customer can have a maximum of one membership.
2. An employee working in a store cannot work more than eight hours a day.
3. Only employees working in a store have shifts.
4. Items in a shipment must have been ordered.
5. The items in an order directed to a supplier must be sold by that supplier.
6. A store’s weekly ads are only sent to members living near the store’s location.
7. A shipment departing from a warehouse should only contain items produced by the supplier that owns the warehouse.
8. The total weight of shipments carried by a truck should not exceed the truck’s capacity.

**Relational Tables:**

EMPLOYEES(eID, LastName, MidInit, FirstName, PhoneNum, Birthday, dID, sID)

DEPENDENTS(eID, Dep\_Name, Sex, Relationship, Birthday)

DEPARTMENT(dID, Department\_name, Street, State, City, Zip)

SHIFT (shID, HourRate, StartTime, EndTime, sID)

STORE (sID, Phone#, Street, State, City, Zip)

MEMBERS (mID, LastName, MidInit, FirstName, Street, City, State, Zip, Phone#, Birthday, sID)

ORDER (oID, Date, sID, supID)

SUPPLIER (supID, supplierName)

WAREHOUSE (wID, Street, City, State, Zip, supID)

SHIPMENT (shipID, Date, wID, sID, plateNo)

TRUCKS(plateNo, capacity, Type)

ITEM(iID, name, weight, supID)

COUPONS(cID, Expiration, Discount, iID)

PRESCRIPTION(pID, Doctor\_name, Patient\_name, sID)

WEEKLY\_AD(adID, sID)

WORKS(eID, shID, Hour)

SENT\_TO(mID, adID)

INCLUDES(adID, cID)

STOCK (sID, iID, Qty)

CONSISTS\_OF(iID, oID)

CONTAINS(iID, shipID)

= Foreign Key

Query 1: Retrieve information about employees (eID, LastName, MidInit, FirstName) that work any shifts or work in stores located in Ohio. (union)

Employees\_Shifts ← π eID, LastName, MidInit, FirstName (EMPLOYEES \* WORKS)

Employees\_Ohio ← π eID, LastName, MidInit, FirstName (σ State = ‘OH’ (EMPLOYEES \* STORE))

Employees\_Shifts ∪ Employees\_Ohio

SELECT e.eID, e.LastName, e.MidInit, e.FirstName

FROM EMPLOYEES AS e

WHERE e.eID IN (SELECT w.eID FROM WORKS AS w)

UNION

SELECT e.eID, e.LastName, e.MidInit, e.FirstName

FROM EMPLOYEES AS e

WHERE e.sID IN (SELECT s.sID FROM STORE AS s WHERE s.State = 'OH')

Query 2: Retrieve information about employees (eID, LastName, MidInit, FirstName) that work any shifts and work in stores located in Ohio. (intersection)

Employees\_Shifts ← π eID, LastName, MidInit, FirstName (EMPLOYEES \* WORKS)

Employees\_Ohio ← π eID, LastName, MidInit, FirstName (σ State = ‘OH’ (EMPLOYEES \* STORE))

Employees\_Shifts ∩ Employees\_Ohio

SELECT e.eID, e.LastName, e.MidInit, e.FirstName

FROM EMPLOYEES AS e

WHERE e.eID IN (SELECT w.eID FROM WORKS AS w)

INTERSECT

SELECT e.eID, e.LastName, e.MidInit, e.FirstName

FROM EMPLOYEES AS e

WHERE e.sID IN (SELECT s.sID FROM STORE AS s WHERE s.State = 'OH')

Query 3: Retrieve the names (first and last) of employees who do not have any dependents.

ALL\_EMPS ← eID (EMPLOYEES)

EMPS\_WITH\_DEPENDENTS ← eID (DEPENDENTS)

RESULT\_IDS ← ALL\_EMPS – EMPS\_WITH\_DEPENDENTS

RESULT\_NAMES ← FirstName, LastName (EMPLOYEES \* RESULT\_IDS)

SELECT e.FirstName, e.LastName

FROM EMPLOYEES AS e

WHERE NOT EXISTS (SELECT \*

FROM DEPENDENTS AS d

WHERE e.eID = d.eID)

Query 4: Retrieve the ID of stores that have every item in stock.

STORES\_ITEMS ← sID, iID (STOCK)

ALL\_ITEMS ← iID (ITEM)

STORES\_ALL\_ITEMS ← STORES\_ITEMS ALL\_ITEMS

SELECT s.sID

FROM STORE AS s

WHERE NOT EXISTS (SELECT i.iID

FROM ITEM AS i

WHERE NOT EXISTS (SELECT \*

FROM STOCK AS t

WHERE i.iID = t.iID AND s.sID = t. sID))

Query 5: Count the number of Warehouse owned by each supplier. Result ordered by warehouse number.

π(supID, supplierName, numWarehouse)

(

(ρCOU(supID, numWarehouse)

(supIDℑCOUNT(wID)(WAREHOUSE))

) \* SUPPLIER

)

select sup.supID, sup.supplierName as 'Supplier Name', cou.numWarehouse as 'Number of Warehouse' from (select supID, count(wID) as numWarehouse from WAREHOUSE group by supID) as cou join SUPPLIER as sup on cou.supID = sup.supID order by numWarehouse desc;

Query 6: Calculate the Payslip of each employee who works in Store. Result ordered by Payslip amount. [Inner Join among EMPLOYEES, WORKS and SHIFT]

π(LastName, MidInit, FirstName, Payslip)

(

(ρS(eID, Payslip)

(eIDℑSUM(Hour \* HourRate)

((σ(sID is not null)(EMPLOYEES\*WORKS))\*SHIFT)

)

) \* EMPLOYEES

)

select em.LastName, em.MidInit, em.FirstName, g.Payslip from EMPLOYEES as em join (select e.eID as eID, sum(w.Hour \* sh.HourRate) as Payslip from EMPLOYEES as e join WORKS as w on e.sID is not null and e.eID = w.eID join SHIFT as sh on w.shID = sh.shID group by e.eID) as g on em.eID = g.eID order by Payslip desc;

Simple Transaction:

Declare @intErrorCode int;

begin tran

update SHIFT set HourRate = 1.05 \* HourRate where HourRate >= 9;

select @intErrorCode = @@ERROR

if(@intErrorCode <> 0) goto Problem

update SHIFT set HourRate = 1.01 \* HourRate where HourRate < 9;

select @intErrorCode = @@ERROR

if(@intErrorCode <> 0) goto Problem

commit tran

Problem:

if(@intErrorCode <> 0) begin

PRINT 'Unexpected error occurred!'

rollback tran

end

Nested Transaction:

SELECT 'Before BEGIN TRAN', @@TRANCOUNT

BEGIN TRAN

SELECT 'After BEGIN TRAN', @@TRANCOUNT

INSERT INTO employees values ('Doe','E','John','6141012929','1968-11-11', NULL, 5)

BEGIN TRAN nested

SELECT 'After BEGIN TRAN nested', @@TRANCOUNT

UPDATE employees

SET sID = 3

WHERE FirstName = ‘John’ AND LastName = ‘Doe’

COMMIT TRAN nested

SELECT 'After COMMIT TRAN nested', @@TRANCOUNT

ROLLBACK TRAN

SELECT 'After ROLLBACK TRAN', @@TRANCOUNT

SELECT \* FROM employees

Transaction with Save Points:

SELECT 'Before BEGIN TRAN', @@TRANCOUNT

-- The value of @@TRANCOUNT is 0

BEGIN TRAN main

SELECT 'After BEGIN TRAN main', @@TRANCOUNT

-- The value of @@TRANCOUNT is 1

INSERT INTO EMPLOYEES values ('Jackson','A','Adam','6149035622','1985-05-16', 1, NULL)

SAVE TRAN addrec

SELECT 'After SAVE TRAN addrec', @@TRANCOUNT

-- The value of @@TRANCOUNT is still 1

SELECT \* FROM EMPLOYEES

BEGIN TRAN nested

SELECT 'After BEGIN TRAN nested', @@TRANCOUNT

-- The value of @@TRANCOUNT is 2

UPDATE EMPLOYEES

SET PhoneNum = '6142245644'

WHERE LastName = 'James' AND FirstName = 'Borg'

SAVE TRAN updaterec

SELECT 'After SAVE TRAN updaterec', @@TRANCOUNT

-- The value of @@TRANCOUNT is still 2

SELECT \* FROM EMPLOYEES

ROLLBACK TRAN addrec

SELECT 'After ROLLBACK TRAN addrec', @@TRANCOUNT

-- The value of @@TRANCOUNT is still 2

SELECT \* FROM EMPLOYEES

IF (@@TRANCOUNT > 0) BEGIN

ROLLBACK TRAN

SELECT 'AFTER ROLLBACK TRAN', @@TRANCOUNT

-- The value of @@TRANCOUNT is 0 because ROLLBACK TRAN always rolls back all transactions and sets @@TRANCOUNT to 0.

END

SELECT \* FROM EMPLOYEES