

# Ingeniería de Software II (CI-4713)

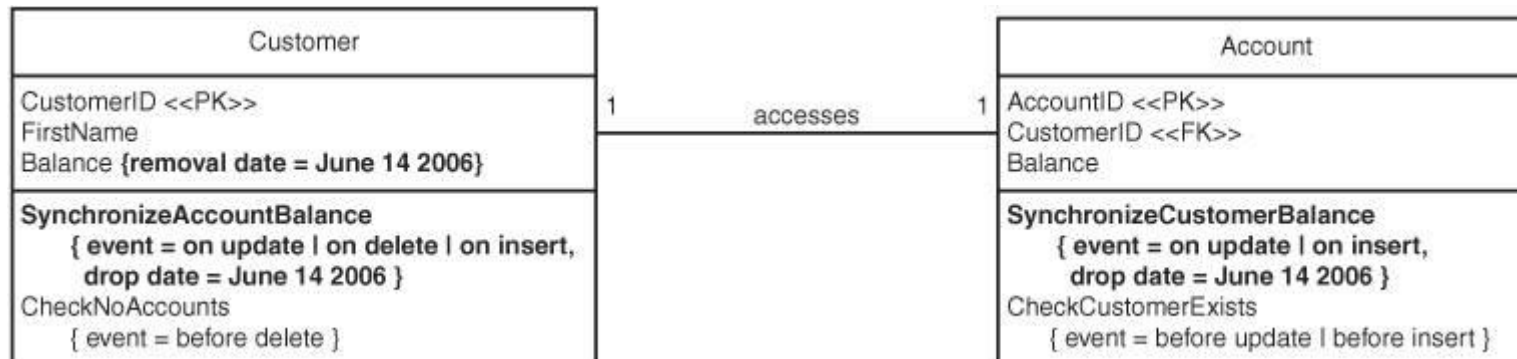
## **Refactorización en Base de Datos**

mayo de 2008

# Refactorización en Bases de Datos

- Un cambio en el esquema de BD para mejorar su diseño manteniendo su semántica y comportamiento.
- Refactorizar aspectos estructurales: definiciones de tablas y vistas.
- Refactorizar aspectos funcionales: *stored procedures* y *triggers*.
- Refactorizaciones sobre BDs son mas difíciles de implementar.
- Los impedimentos son del tipo cultural y carencia de herramientas.

# Refactorización en Bases de Datos



# Refactorización en Bases de Datos

```
ALTER TABLE Account ADD Balance Numeric;
```

```
CREATE OR REPLACE TRIGGER  
    SynchronizeCustomerBalance  
BEFORE INSERT OR UPDATE ON Account  
REFERENCING OLD AS OLD NEW AS NEW  
FOR EACH ROW  
DECLARE  
BEGIN  
IF :NEW.Balance IS NOT NULL THEN  
    UpdateCustomerBalance; END IF;  
END;
```

# Refactorización Estructural

- Drop Column
  - Mecanismos de actualización de esquemas:
    - Opción DROP COLUMN de ALTER TABLE o creación de una tabla temporal.
    - Opción SET UNUSED de ALTER TABLE
    - Si la columna es clave principal, removerla de las claves foráneas asociadas y reemplazar la clave primaria en la tabla.
  - Mecanismos de migración
    - CREATE TABLE CustomerFavoriteColor AS SELECT CustomerID, FavoriteColor FROM Customer;

# Refactorización Estructural

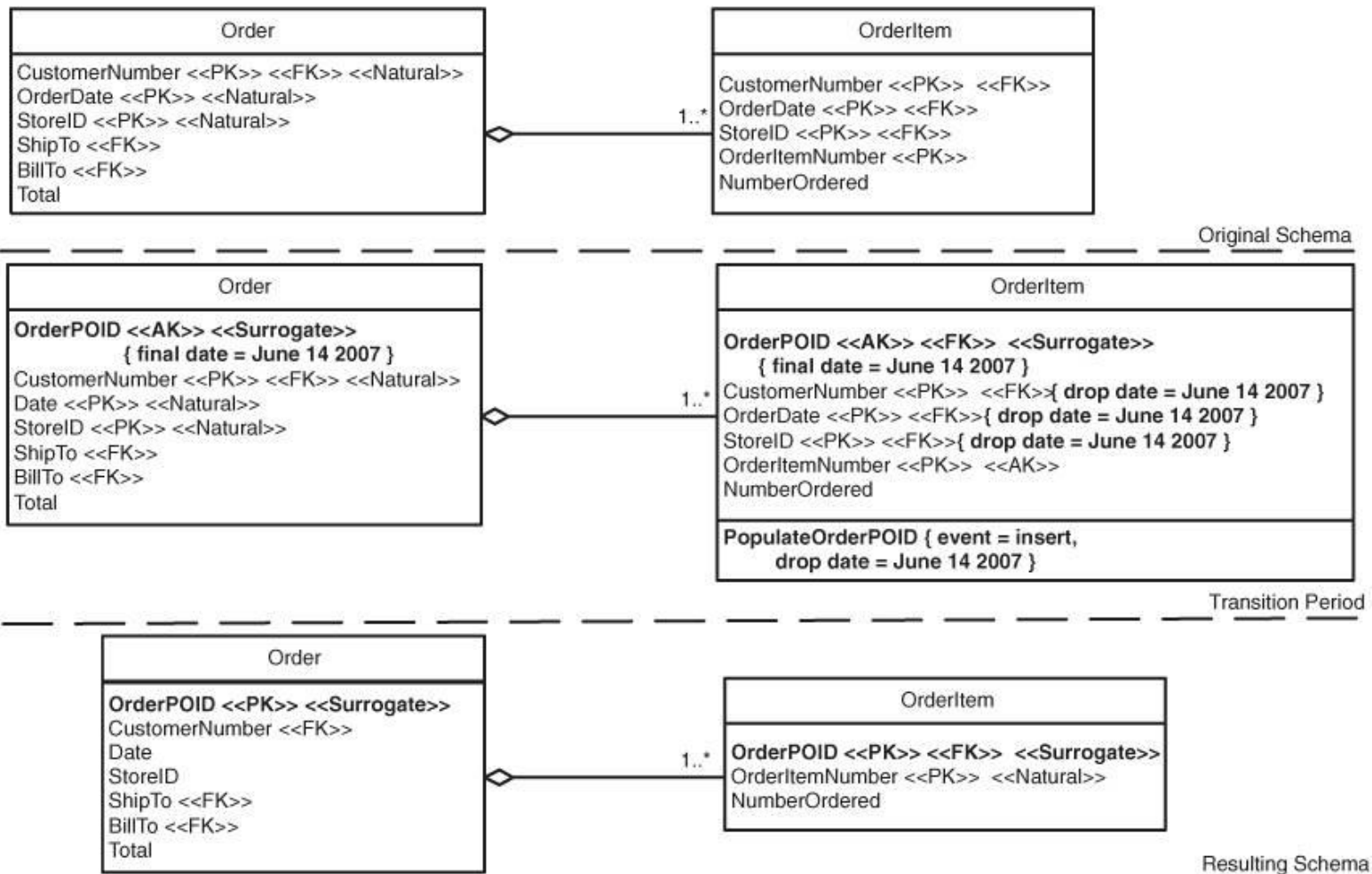
- Drop View
  - Mecanismos de actualización de esquemas:
    - DROP VIEW.
- Drop Table
  - Mecanismos de actualización de esquemas:
    - Remover restricciones de claves foráneas hacia la tabla a eliminar
    - Renombrar la tabla y automáticamente se cambian todas las referencias hacia la tabla renombrada
  - Mecanismos de migración:
    - CREATE TABLE TaxJurisdictionsRemoved AS SELECT \* FROM TaxJurisdictions;

# Refactorización Estructural

- Introduce Calculated Column
  - Mecanismos de actualización de esquemas:
    - Determinar estrategia de sincronización: batch jobs, actualizaciones, triggers.
    - Determinar el cálculo del valor.
    - Determinar cuál tabla contendrá la columna.
    - Añadir la columna
    - Implementar la estrategia
  - Mecanismos de migración:
    - `UPDATE Customer SET TotalAccountBalance = (SELECT SUM(balance) FROM Account WHERE Account.CustomerId = Customer.CustomerId)`

# Refactorización Estructural

- Introduce Surrogate Key

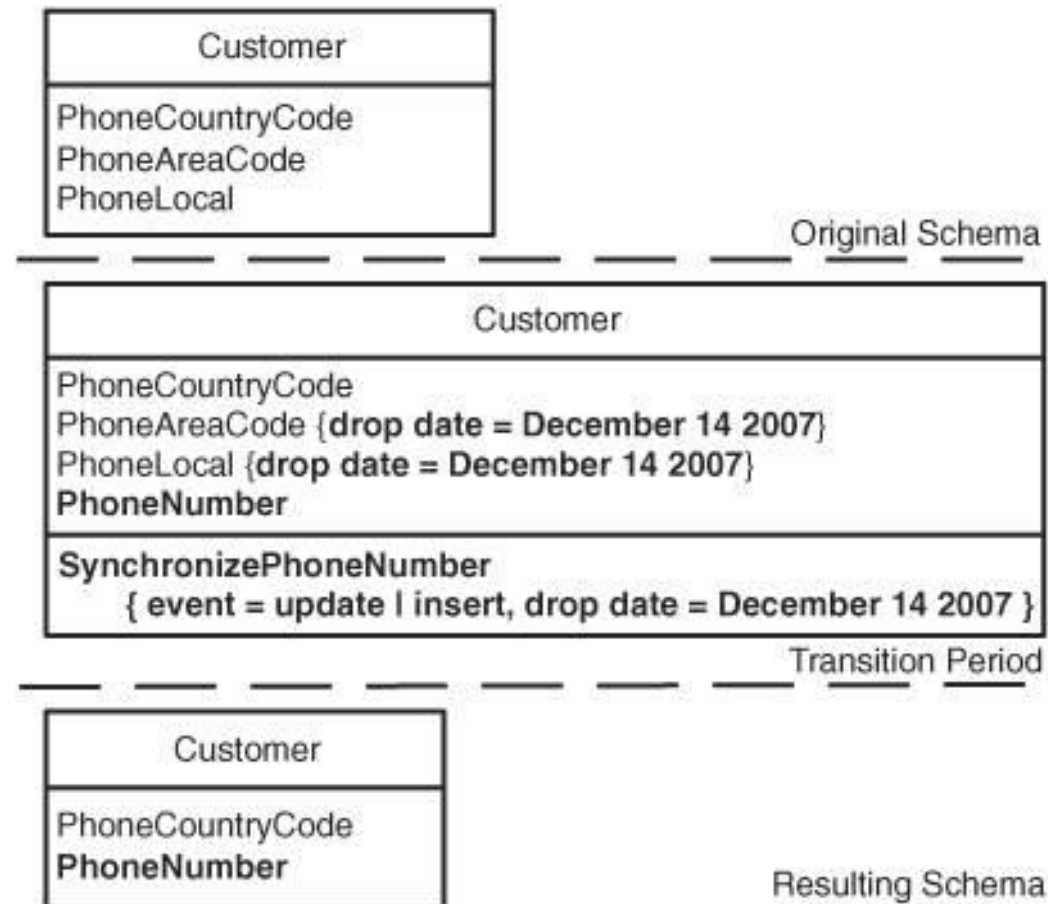


# Refactorización Estructural

- Introduce Surrogate Key (Inverso a Replace Surrogate Key with Natural Key)
  - Mecanismos de actualización de esquemas:
    - Añadir la nueva columna (ADD COLUMN)
    - Añadir un nuevo índice para la clave.
    - Deprecate la columna original
    - Añadir y actualizar los triggers de integridad referencial.
  - Mecanismos de migración:
    - Generar valores para la clave y asignar esos valores a las claves foráneas

# Refactorización Estructural

- Merge Columns (Inverso Split Column)



# Refactorización Estructural

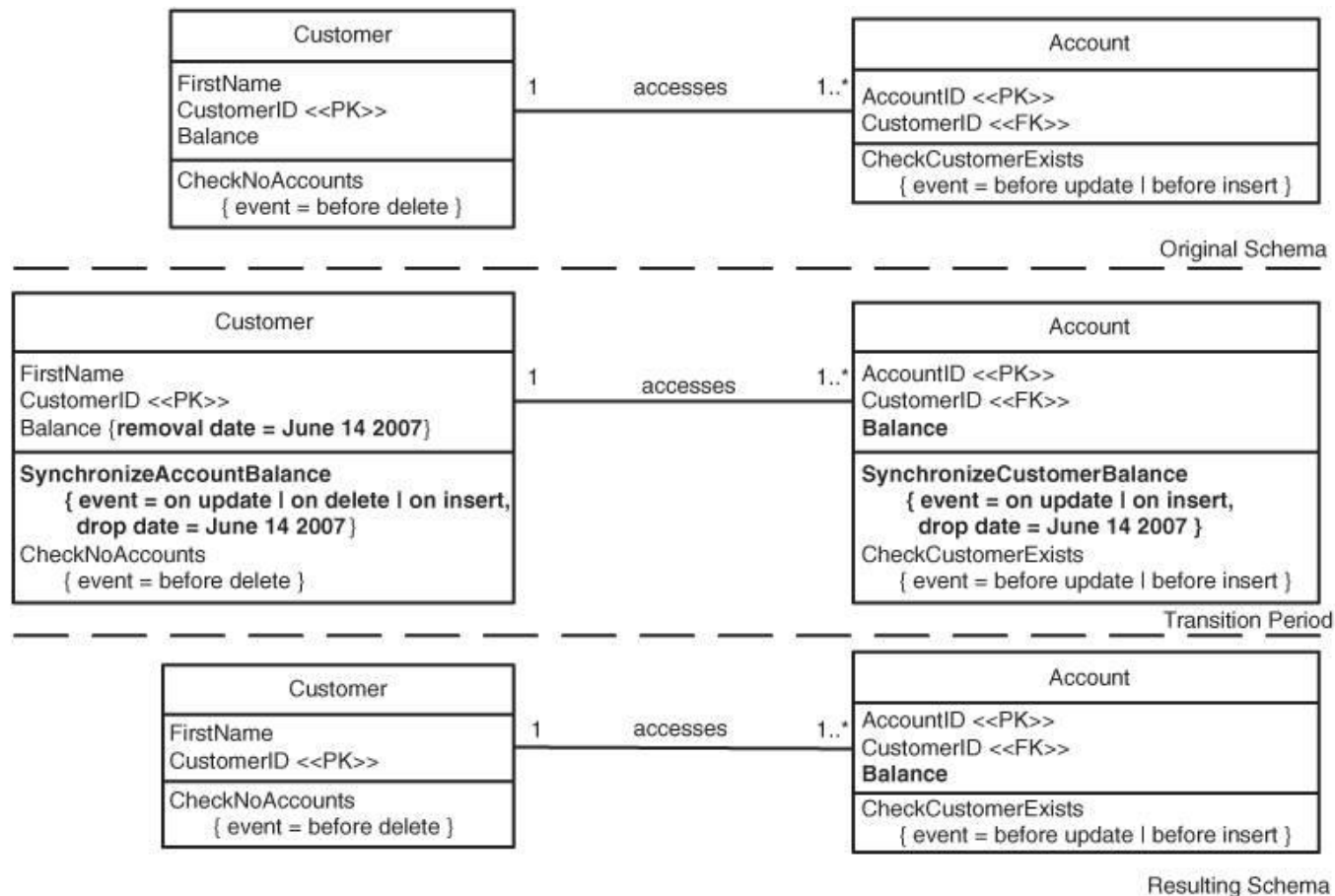
- Merge Columns
  - Mecanismos de actualización de esquemas:
    - Añadir la nueva columna (ADD COLUMN)
    - Implementar un trigger de sincronización entre las columnas.
  - Mecanismos de migración:
    - UPDATE Customer SET PhoneNumber = PhoneAreaCode\*10000000 + PhoneLocal);

# Refactorización Estructural

- Merge Tables (Inverso Split Tables)
  - Mecanismos de actualización de esquemas:
    - Crear la tabla que será la mezcla de las dos tablas.
    - Implementar triggers de sincronización entre las tres tablas (Evitar triggers ciclicos).
  - Mecanismos de migración:
    - Mediante un script o carga de datos

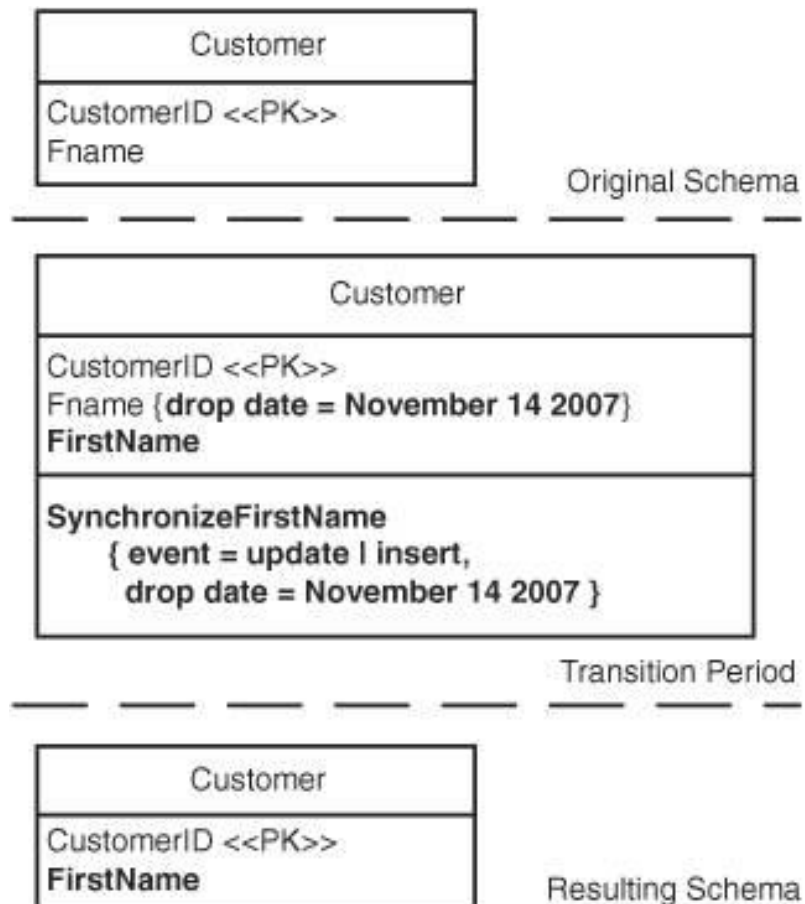
# Refactorización Estructural

- Move Column



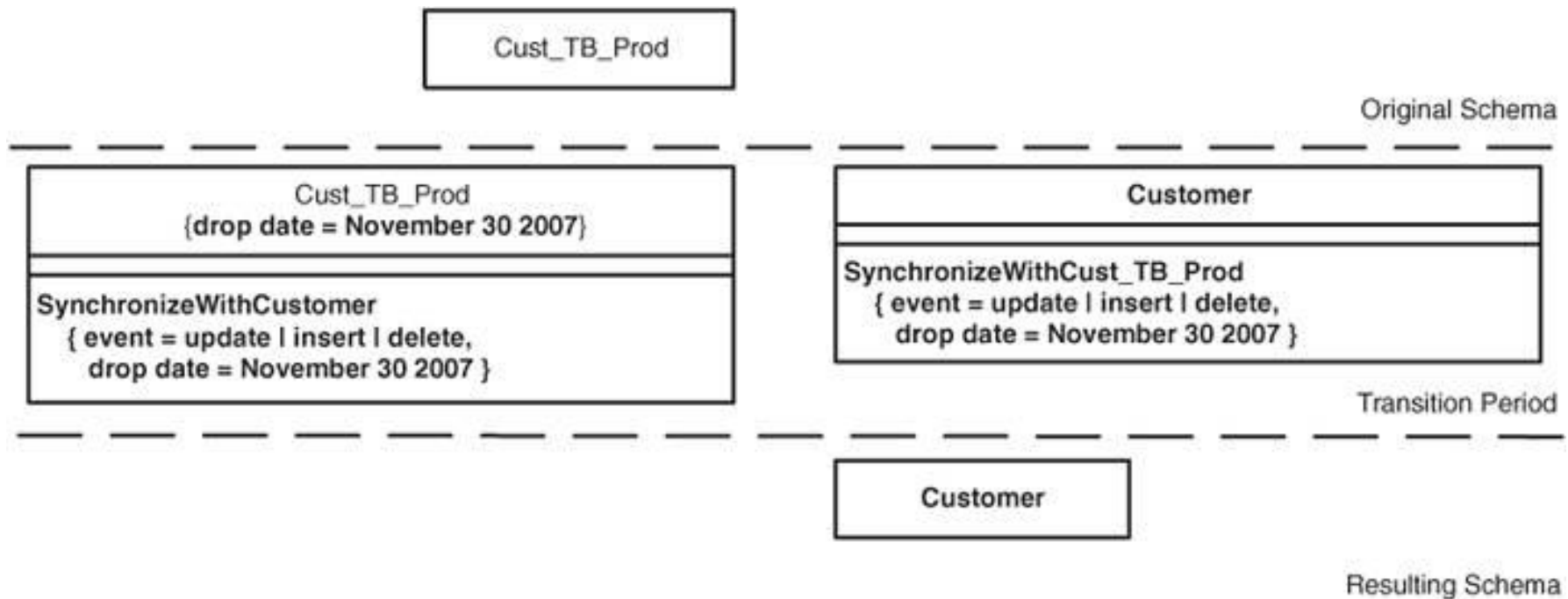
# Refactorización Estructural

- Rename Column



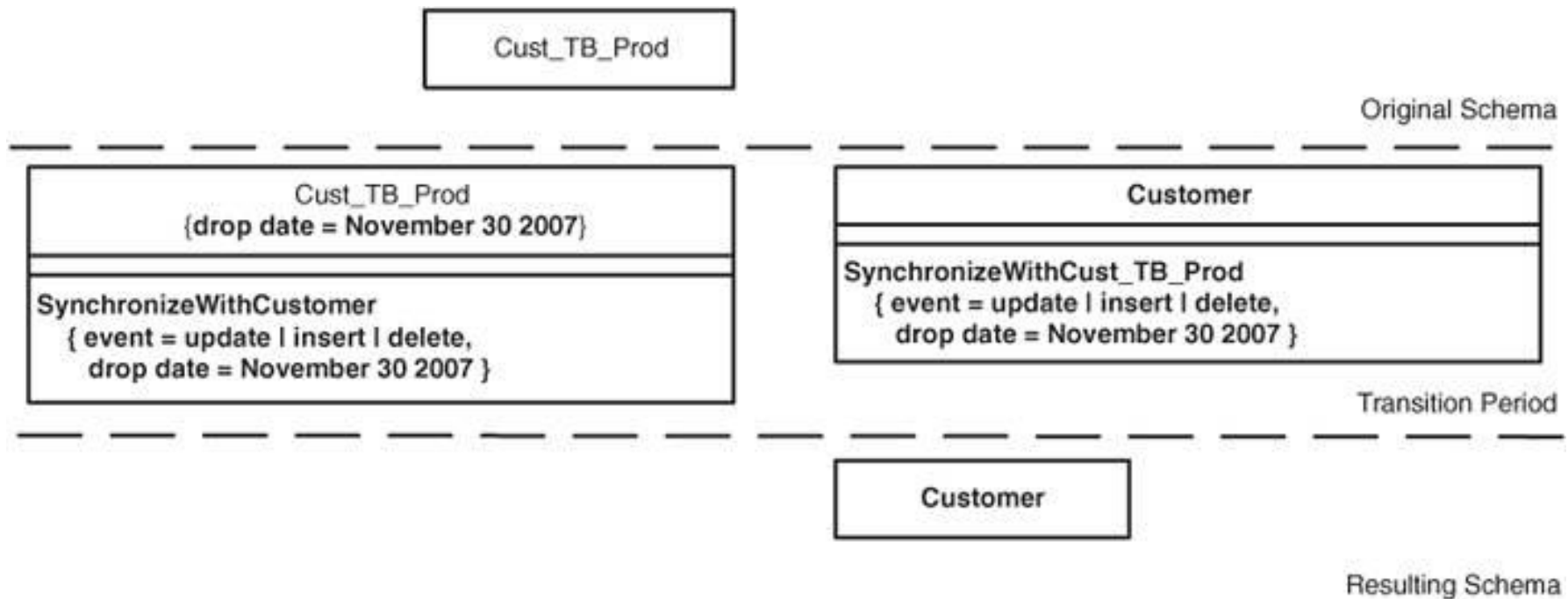
# Refactorización Estructural

- Rename Table



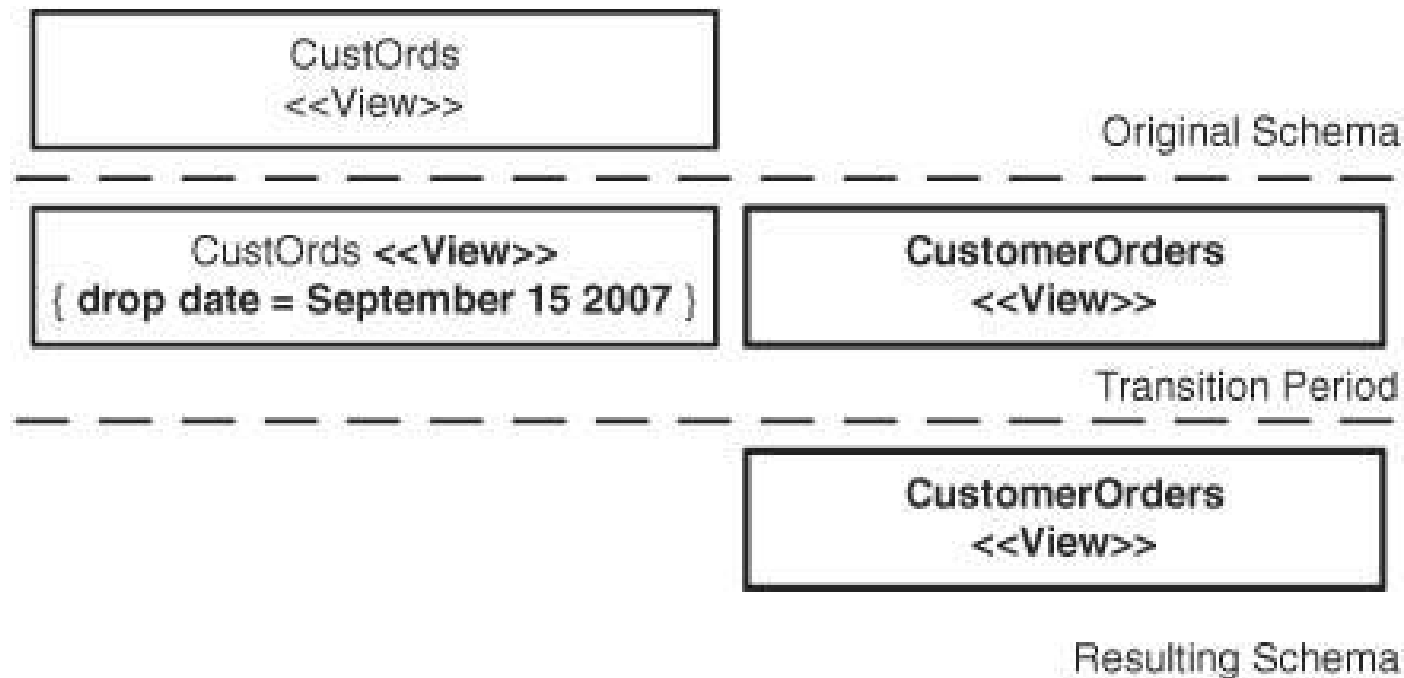
# Refactorización Estructural

- Rename Table



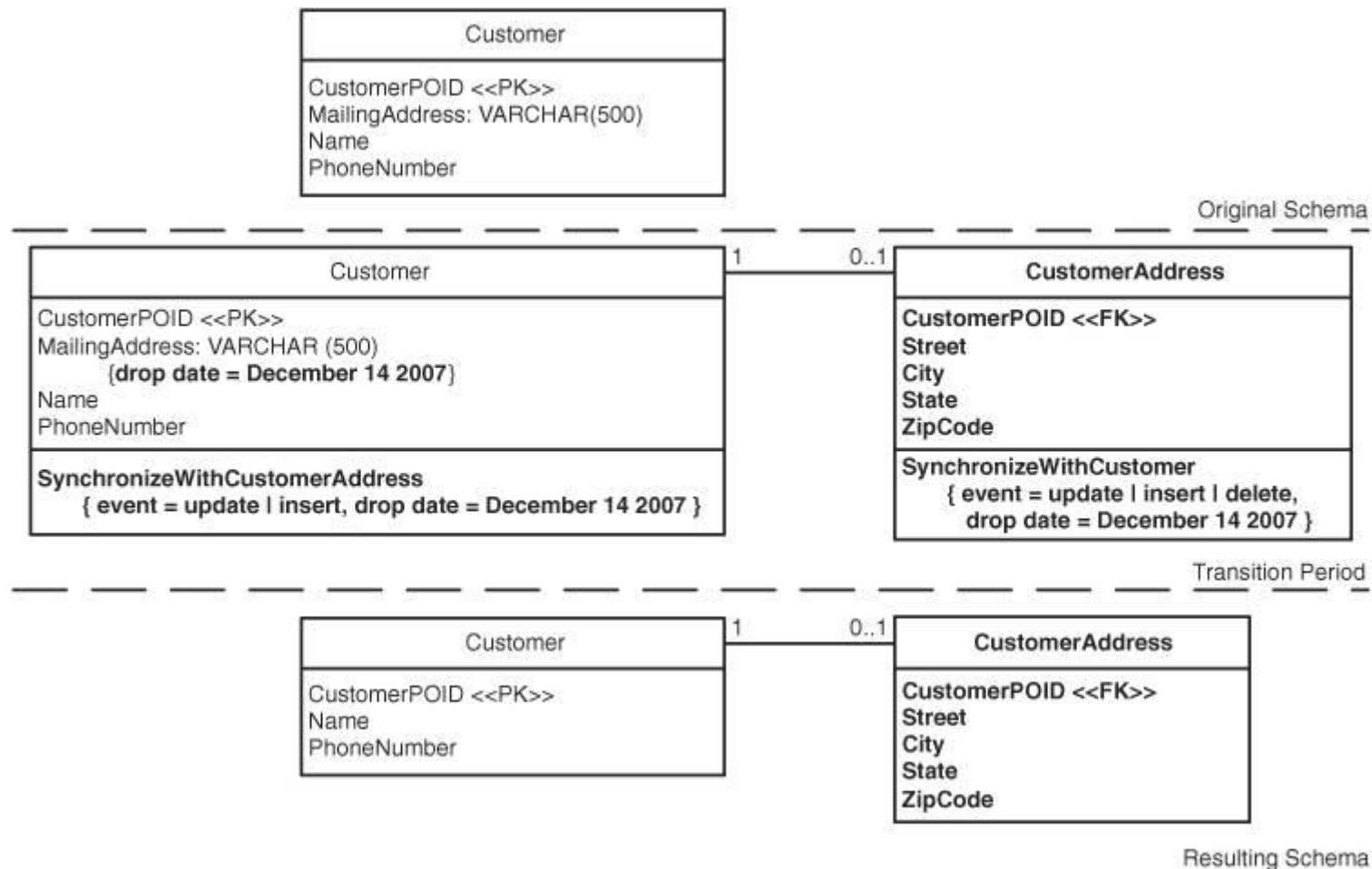
# Refactorización Estructural

- Rename View



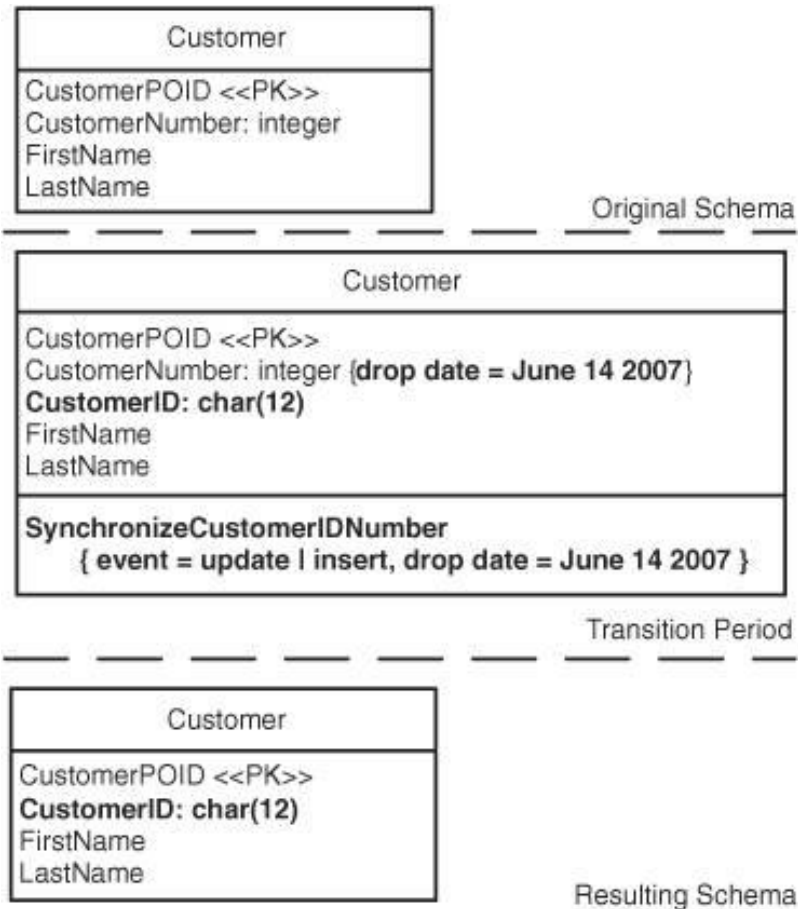
# Refactorización Estructural

- Replace LOB with Table



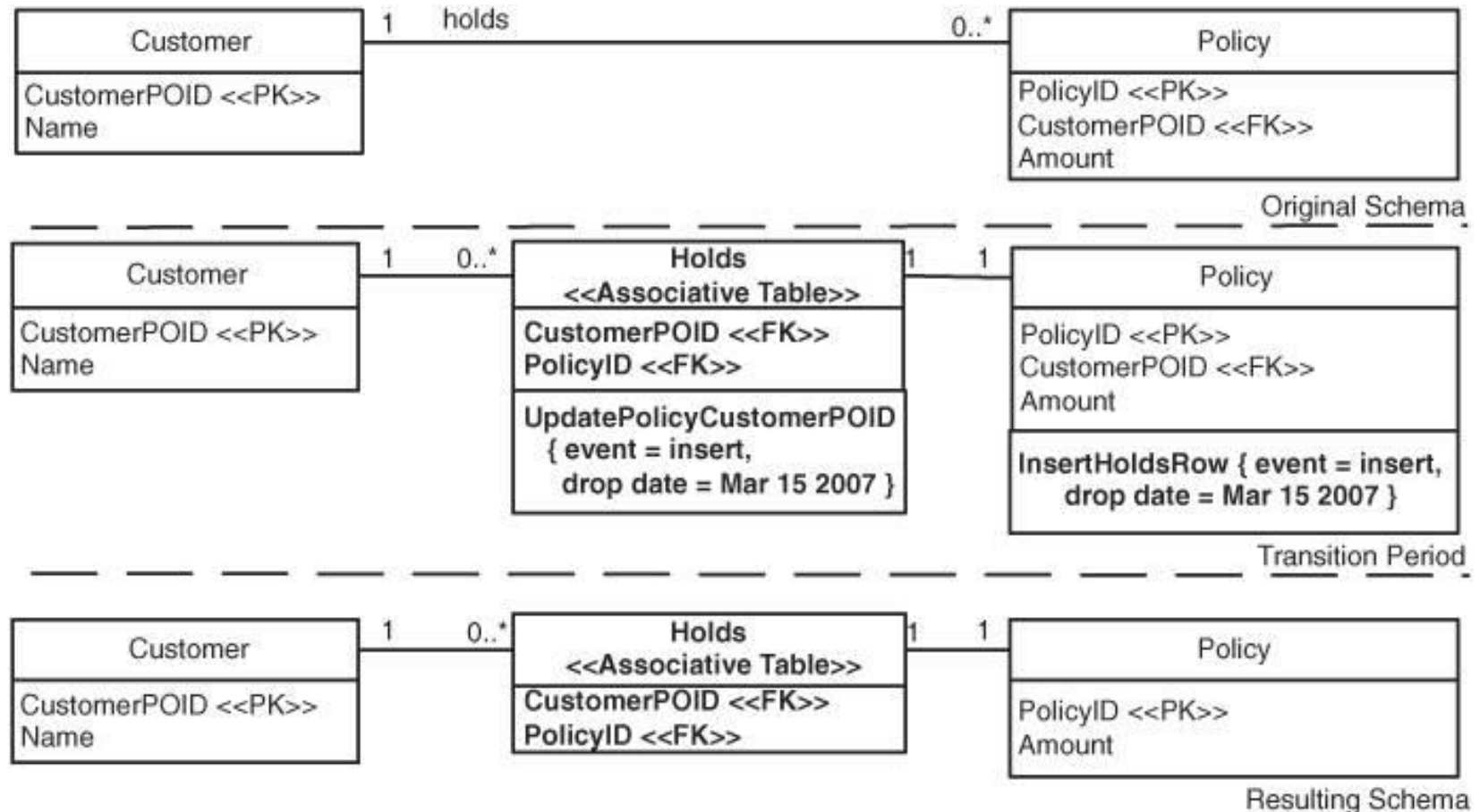
# Refactorización Estructural

- Replace Column



# Refactorización Estructural

- Replace 1:N with Associative Table



# Referencias

- **Refactoring Databases. Evolutionary Database Design. Ambler, S. and Sadalage, P. Addison Wesley 2006**