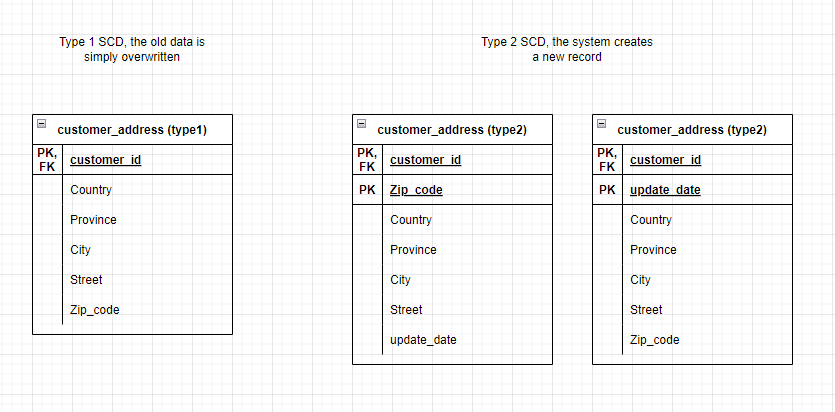
**Question 3**

The store wants to keep customer addresses. Propose two architectures for the CUSTOMER\_ADDRESS table, one that will retain changes, and another that will overwrite. Which is type 1, which is type 2?

*Hint, search type 1 vs type 2 slowly changing dimensions.*

Bonus: Are there privacy implications to this, why or why not?



**Type 1:** In Type 1 SCD, the old data is simply overwritten with the new data. This means that historical data is not preserved. On the first ERD table "customer\_address (type1)," we have only one primary key "customer\_id," which is a unique value in the table, so we can have only one record for each customer. If we need to add another address to an existing row in "customer\_address (type1)," we can only rewrite it as we prohibit having duplicate values for the primary key.

We can make "zip\_code" instead of "customer\_id" as the PK. The architecture will remain the same. This kind of architecture fits better for non-changing dimensions or if we don't need to keep historical changes or logs.

**Type 2:** In Type 2 SCD, the system creates a new record when a change occurs, preserving historical data. The second and third ERD tables "customer\_address (type2)" have two primary keys, so it keeps a unique value through the pair of columns "customer\_id" – "zip\_code" and "customer\_id" – "update\_date." In this case, each customer can have many unique zip\_codes.

This architecture is better for cases where it is required to save historical changes or logs.

**Bonus:** Type 1 SCD architecture can be considered when the privacy concept is implemented. Decreasing privacy data in the DB decreases attraction to use this data in inappropriate ways, and also, a lesser amount can be leaked.

**Question 4**

Review the AdventureWorks Schema [here](https://i.stack.imgur.com/LMu4W.gif)

Highlight at least two differences between it and your ERD. Would you change anything in yours?

* In my schema, each table represented in "AdventureWorks" acts as a topic for a bunch of related tables. The "AdventureWorks" schema is much more diverse and detailed, definitely.
* There are some tables that exist separately and don't relate to any other table. For example, "TransactionHistoryArchive" and "ErrorLog".
* Customers and employees share one table called "person". Additional information regarding customers or employees is expanded in other tables.

I would like to change the architecture in the "Book store", especially by avoiding repeating columns in customers and employees, and instead keeping similar information in one separate table like "person", then expanding the info in other tables.