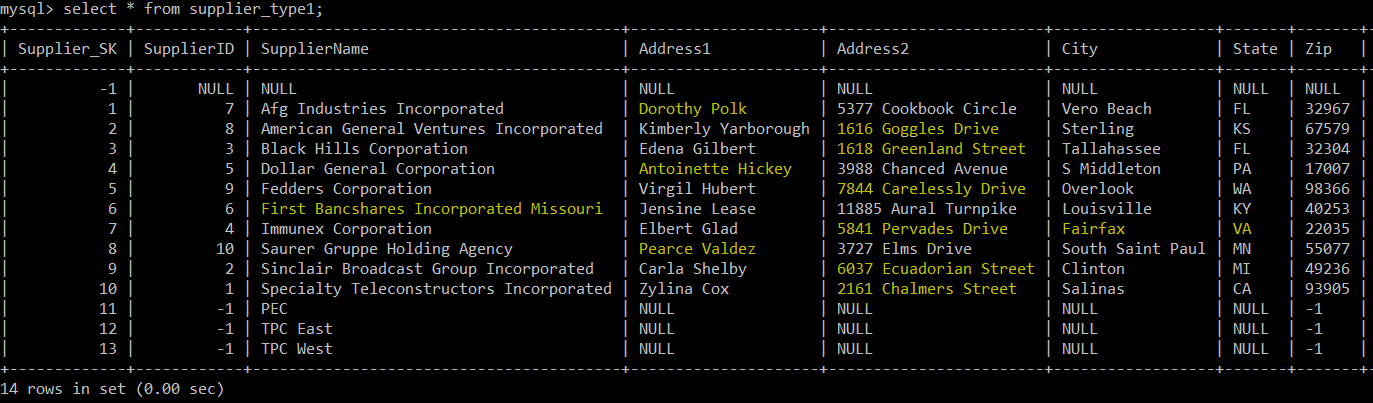
**SCD TYPE 1 (Overwrite) – On supplier dimension**

Slowly Changing Dimension type 1 is for overwriting existing data, which is present in the table. Generally, these data are erroneous data and occurs because of typo error or wrong information present. Since, these data are error-prone data so there is no need to maintain its history.

As a part of our SCD type1 implementation, we have chosen supplier table. We have assumed that there are some wrong entries (cell values) present in the attribute such as SupplierName, Address1, Address2, City, State, and Zip, which is required to be updated. Below is the screenshot of the supplier\_type1 table (the table which we have copied from existing supplier dimension table) before implementation of SCD type1.

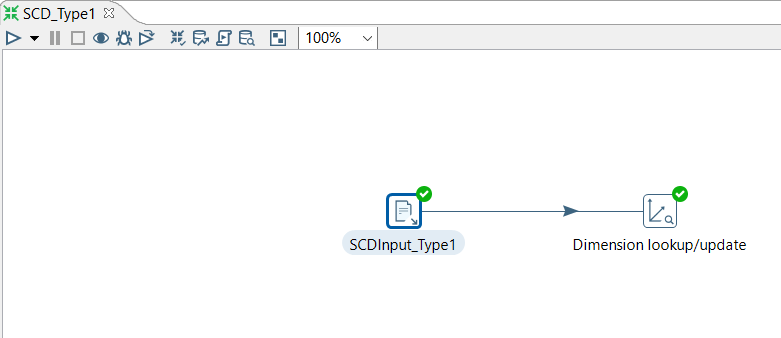
**Before SCD:** Supplier\_type1

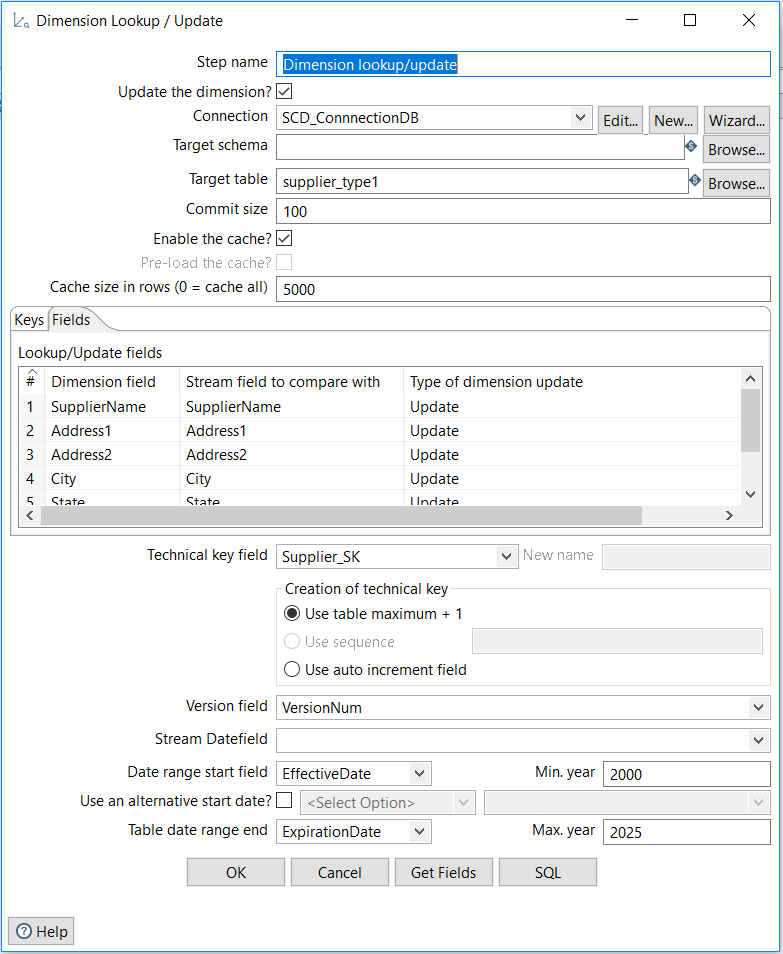


**Implementation (KTR):**

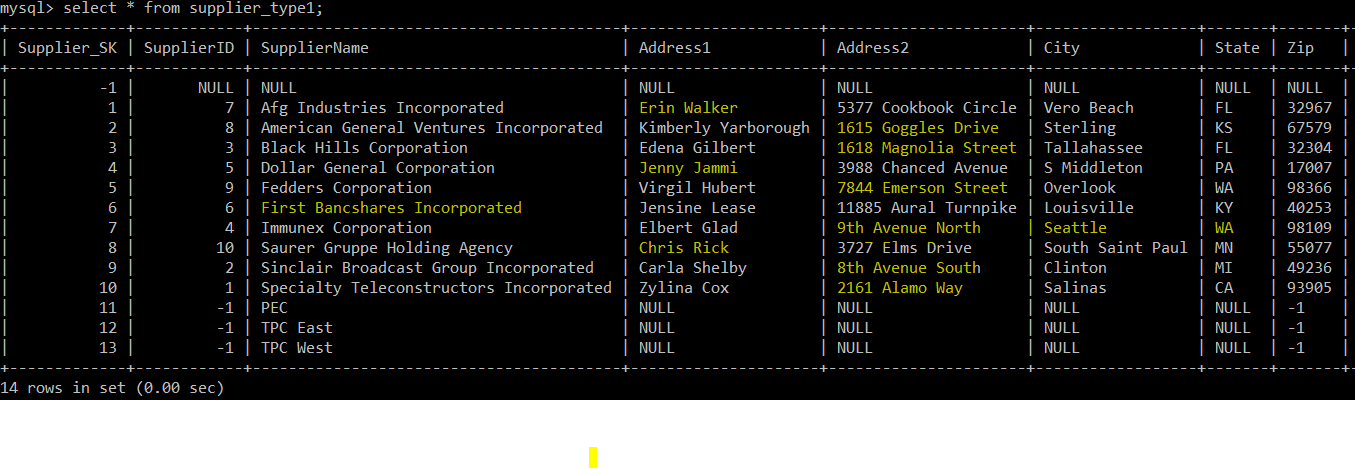
We have created individual SCDInput\_Type1.CSV file for updated values. In addition, we have implemented a process in the SCD\_Type1.ktr file. The implemented process reads the input CSV file, which has updated attribute value for SupplierName, Address1, Address2, City, and Zip. Followed by this, the process also does a dimension lookup for all the rows based on SupplierID and update the attribute's value. This is achieved by specifying dimension field, stream field to compare with, and type of dimension to update. For the field attribute that is required to update, type of dimension to update has a value of update and for others, it has a value of punch-through. We also need to specify a surrogate key field as a key field. After populating the updated value in the table dimension lookup/update activity creates additional column that contains VersionNum, EffectiveDate, and ExpirationDate, which is not required to maintain as a part of SCD Type1. So, we manually removed these rows using alter table command.

ALTER TABLE supplier\_type1 DROP VersionNum, DROP EffectiveDate, DROP ExpirationDate;





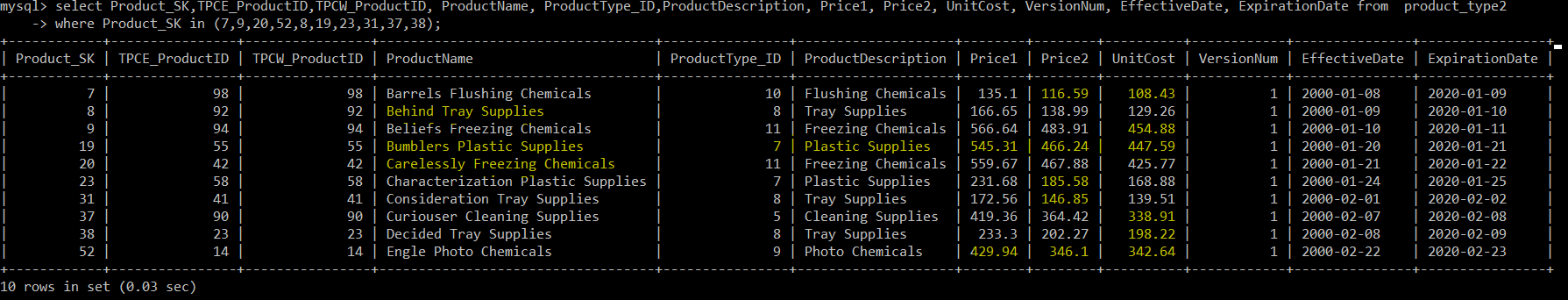
**After SCD:** Supplier\_type1



**SCD TYPE 2 (Add a new row) – On product dimension**

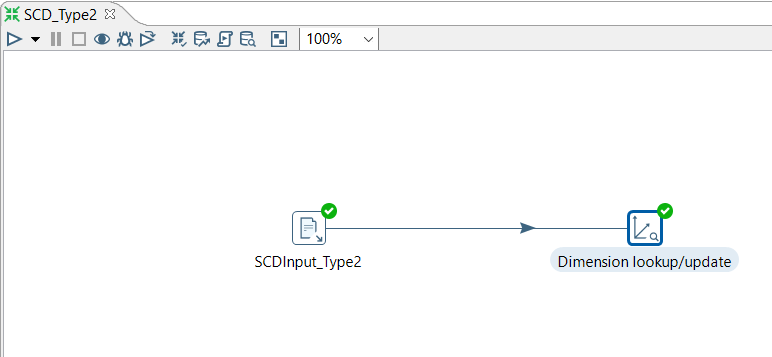
**Before SCD:** Product\_type2

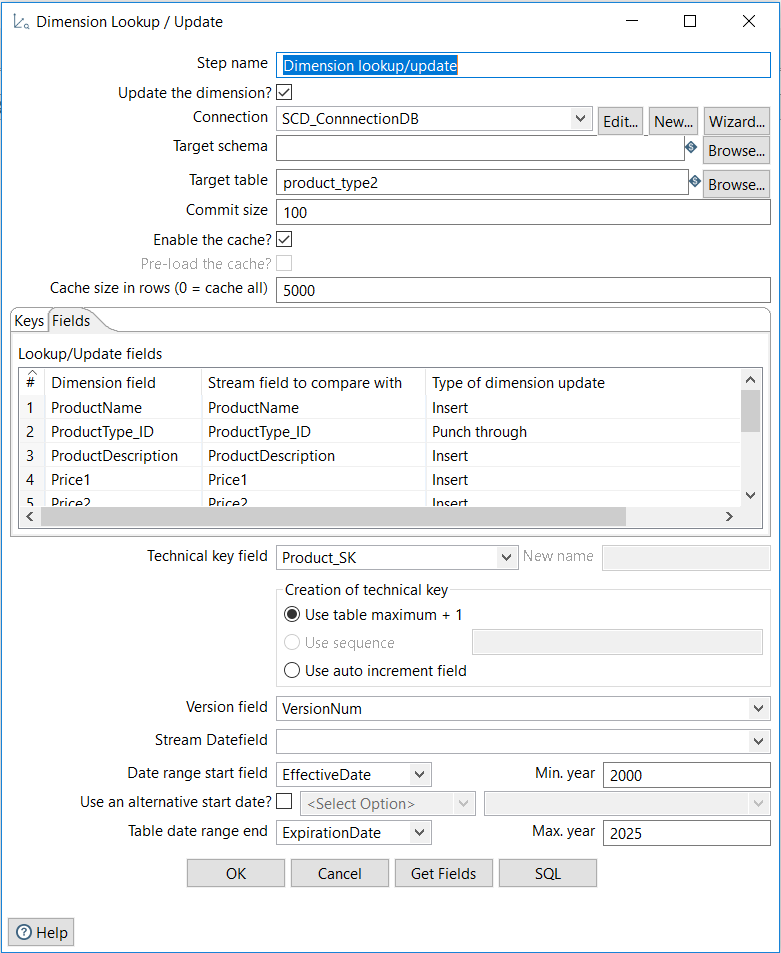
Slowly changing dimension type 2 is for creating new entries. This happens in the case when an error change has occurred that is relevant to the business rules of DW and it is important to preserve the history of records. Hence, a new row with new surrogate key needs to be entered with the new changes for each of the records. For all the record before changes must reflect the old value and all the record after subsequent changes must reflect a new value. In addition, we have to maintain subsequent version number effective date and expiration date of all the records in this case. For SCD type2 implementation we have chosen product dimension.



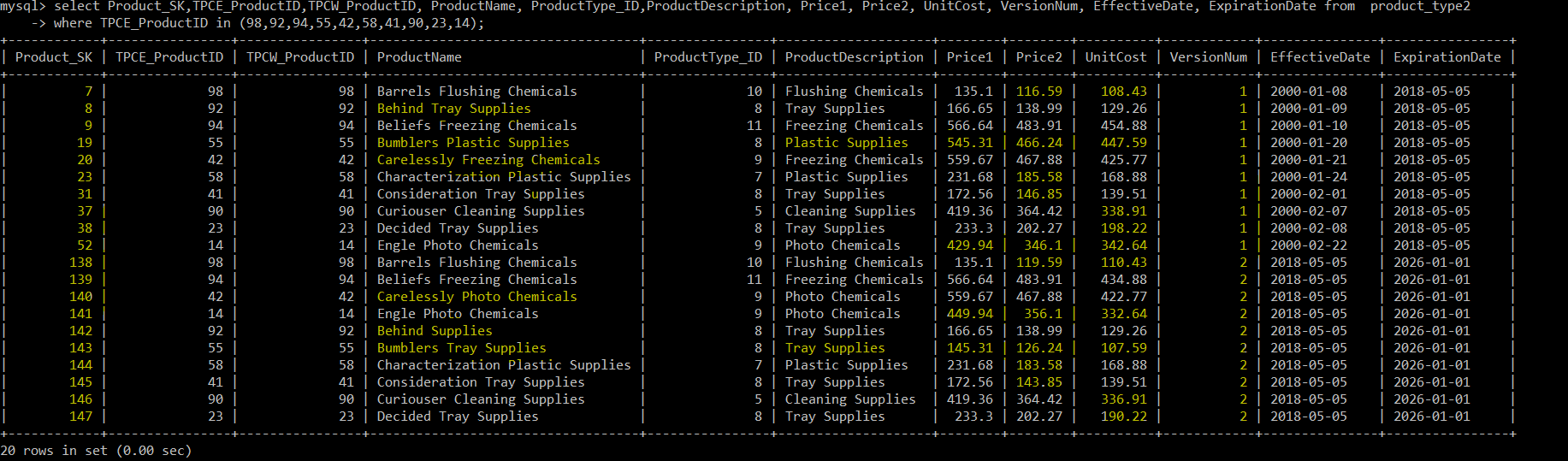
**Implementation (KTR):**

As a part of the implementation, we created a SCDInput\_Type2.CSV file that contains changes in the attribute values of ProductName, ProductDescription, Price1, Price2, and UnitCost. Since we have to insert a new row for each record, so we need to specify Type of dimension as an insert for those attributes whose values have been changed rather than update or punch through. For all the other attributes we have to mention punch through in the Type of dimension field. In addition, we need to do a lookup based upon division ids such as TPCE\_ProductID and TPCW\_ProductID. In the key field, we have to specify a technical key field as Product\_SK. For maintaining VersionNum, EffectiveDate, and ExpirationDate we have to specify these fields in the Dimension lookup/update activity. One important thing about Dimension lookup/update activity is that when it finds a change in the record it creates a new record with a new surrogate key, modifies the expiration date of old record and inserts effective date of new record accordingly.





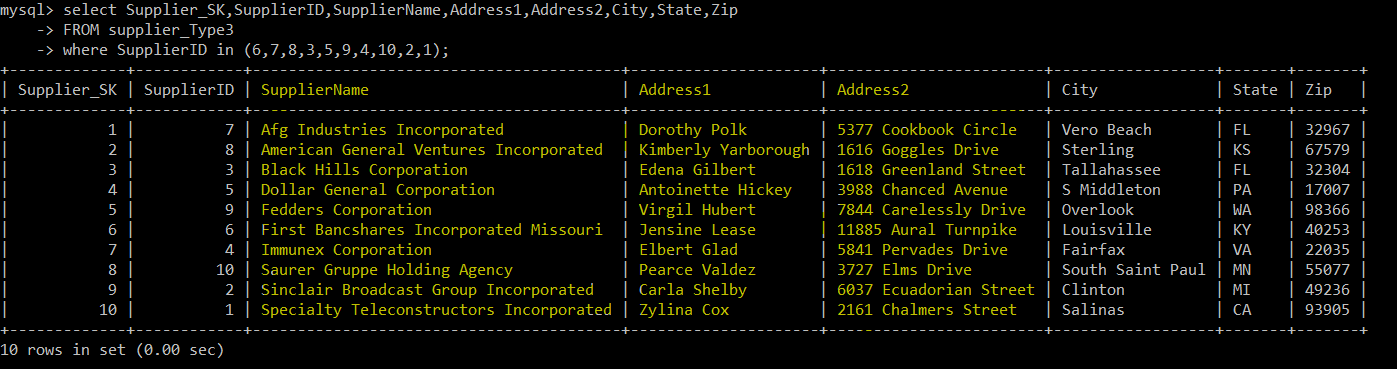
**After SCD :** Product\_type2



**SCD TYPE 3 (Add New Attribute) – On supplier dimension**

Slowly changing dimension type 3 is for tentative or soft changes. In this case, history before and after a change needs to be preserved. In addition, for each of the subsequence changes all the old values needs to be maintained. Also, sometimes it is required to maintain effective date and expiration date for the new attribute values. However, there is no need to maintain any version number. In SCD type3, only a new column required to add for each attribute changes, so surrogate key remains the same. For our, SCD type3 implementation we have chosen Supplier dimension once again.

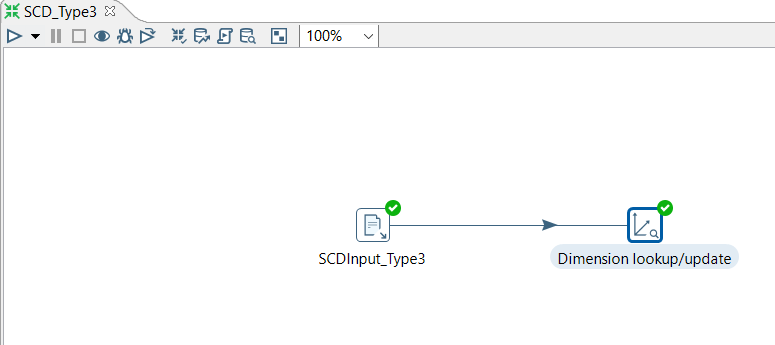
**Before SCD:** Supplier\_type3

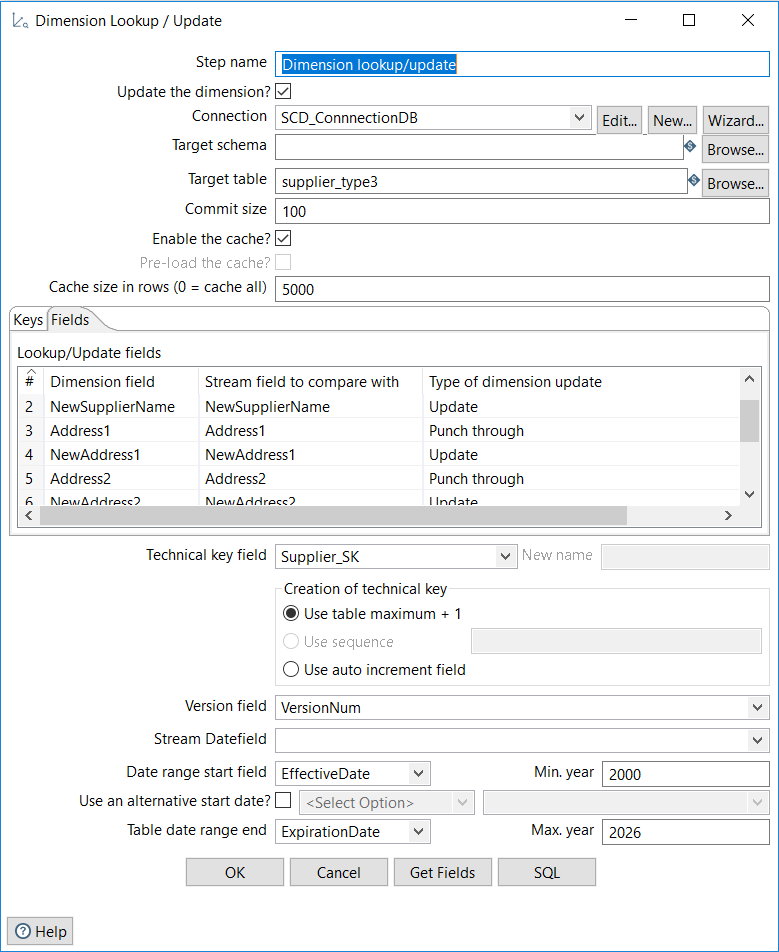


**Implementation (KTR):**

As a part of SCD Type3 implementation, we have created input SCDInput\_Type3.CSV file with additional columns for the changed attribute values. We assumed that some suppliers are acquired by the new organization, so their name and address has changed. Hence, we have created three additional column NewSupplierName, NewAddress1, and NewAddress2. In addition, we have maintained EffectiveDate and ExpirationDate that shows when the update happened and till when it is valid. However, we are not maintaining any VersionNum. So, we need to remove the field from the final table. Additionally, We have configured the process to lookup upon SupplierID. Further, for all the old attribute we have configured Type of dimension update field with a punch through whereas for new attribute values we have configured Type of dimension update field with an update. Lastly, we have to configure for EffectiveDate and ExpirationDate fields.

ALTER TABLE supplier\_type3 DROP VersionNum;





**After SCD:** Supplier\_type3



