

| Business Template  **Loading DM dimensions** |
| --- |
|  |

----Test that the procedure can be executed repeatedly with consistent results--------------------

--------------------------------------------------------------------------------------------------------------------------

---1.1.Run the procedure:

----------------------------------------------------------------------------------------------------------------------------------

**CALL** bl\_cl.sp\_run\_batch\_etl\_by\_day(**DATE** '2022-02-06', **DATE** '2022-02-07');

----------------------------------------------------------------------------------------------------------------------------------------

---1.2.Query the logging table for affected rows:

-------------------------------------------------------------------------------------------------------------------------------------------------

**SELECT** \*

**FROM** bl\_3nf.etl\_logs

**WHERE** procedure\_name **IN** (

'sp\_upsert\_lkp\_game\_types',

'sp\_upsert\_lkp\_game\_categories',

'sp\_upsert\_lkp\_game\_numbers',

'sp\_upsert\_lkp\_payment\_methods',

'sp\_upsert\_lkp\_states',

'sp\_upsert\_lkp\_cities',

'sp\_upsert\_lkp\_zips',

'sp\_upsert\_lkp\_location\_names',

'sp\_upsert\_lkp\_retailers',

'sp\_upsert\_lkp\_statuses',

'sp\_upsert\_lkp\_departments',

'sp\_upsert\_lkp\_employees',

'sp\_upsert\_lkp\_customers',

'sp\_upsert\_lkp\_sales',

'p\_load\_ce\_game\_types',

'p\_load\_ce\_game\_categories',

'p\_load\_ce\_game\_numbers',

'p\_load\_ce\_payment\_methods',

'p\_load\_ce\_states',

'p\_load\_ce\_cities',

'p\_load\_ce\_zip',

'p\_load\_ce\_location\_names',

'p\_load\_retailer\_license\_numbers',

'p\_load\_ce\_statuses',

'p\_load\_ce\_departments',

'p\_load\_ce\_employees',

'p\_load\_ce\_customers\_scd',

'p\_load\_ce\_sales',

'sp\_upsert\_dim\_game\_numbers',

'sp\_upsert\_dim\_payment\_methods',

'sp\_upsert\_dim\_retailer\_license\_numbers',

'sp\_upsert\_dim\_employees',

'sp\_upsert\_dim\_customers\_scd',

'sp\_insert\_fct\_sales'

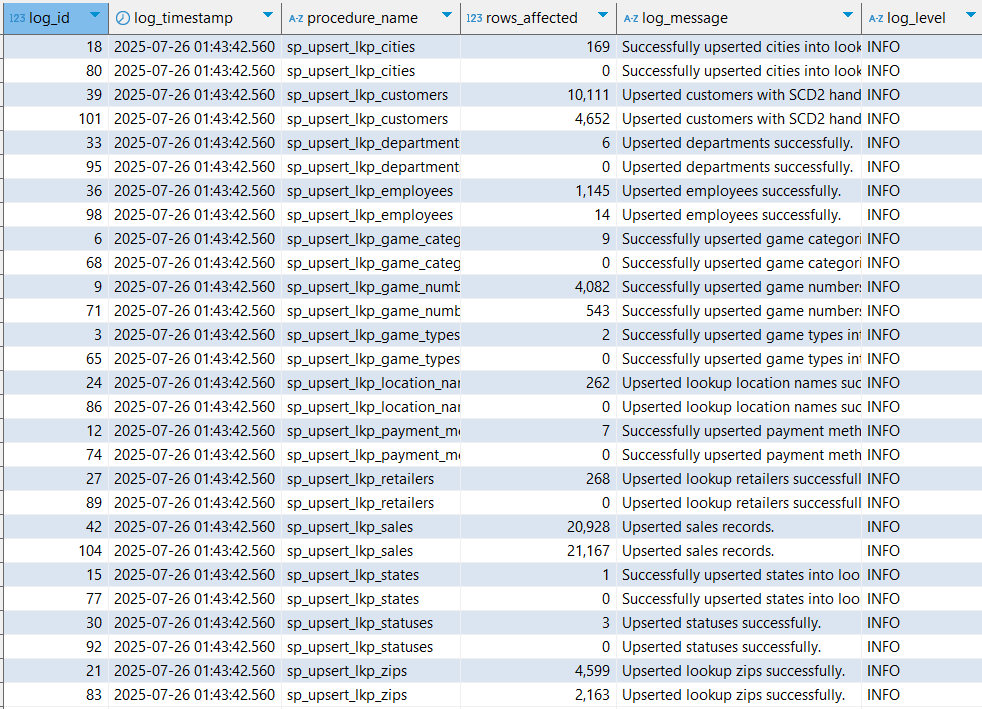
)

**order** **by** procedure\_name, log\_id ;

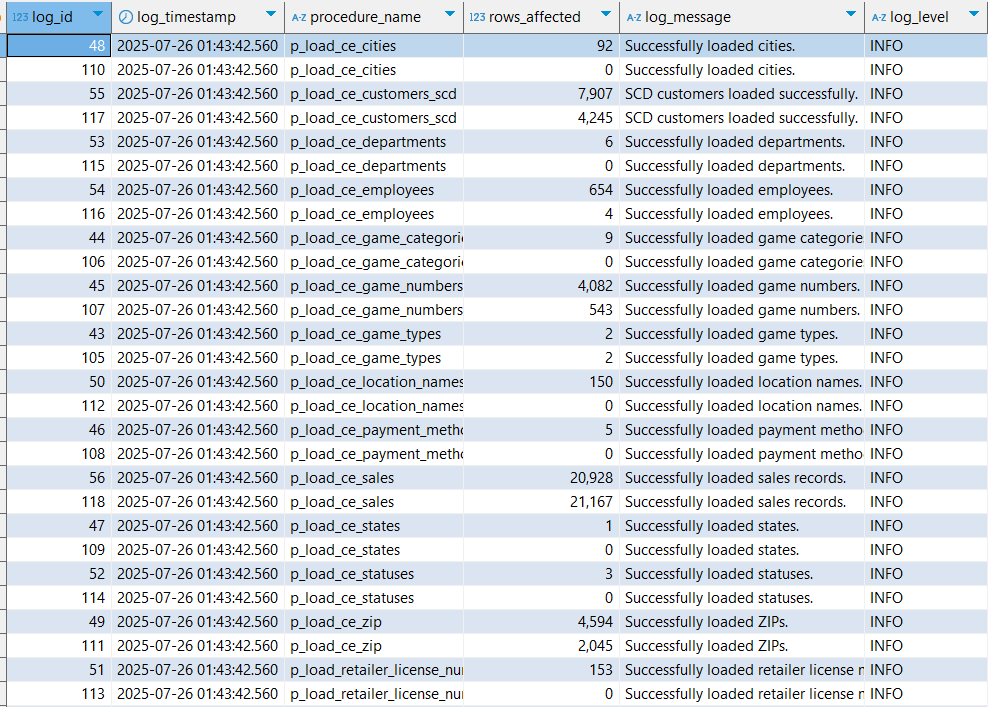
----------------------------------------------------------------------------------------------------------------------------------------

---1.3.Take a screenshot of the logging result

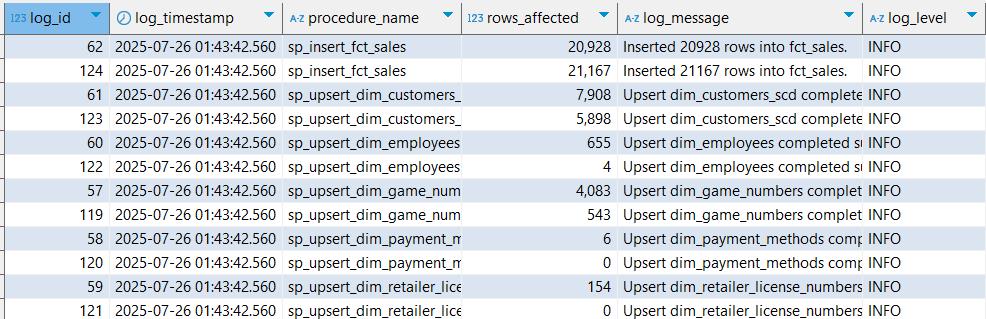
-----------------------------------------------------------------------------------------------------------------------------------------------



The procedures for the lookup insertion tables



The procedures for the 3nf load tables.



The procedures for the dm load tables

----------------------------------------------------------------------------------------------------------------------------------------

---1.4. Run the procedure again with the same input

--------------------------------------------------------------------------------------------------------------------------------------------------

**CALL** bl\_cl.sp\_run\_batch\_etl\_by\_day(**DATE** '2022-02-06', **DATE** '2022-02-07');

----------------------------------------------------------------------------------------------------------------------------------------

---1.5. Query the login table again

--------------------------------------------------------------------------------------------------------------------------------------------------

**SELECT** \*

**FROM** bl\_3nf.etl\_logs

**WHERE** procedure\_name **IN** (

'sp\_upsert\_lkp\_game\_types',

'sp\_upsert\_lkp\_game\_categories',

'sp\_upsert\_lkp\_game\_numbers',

'sp\_upsert\_lkp\_payment\_methods',

'sp\_upsert\_lkp\_states',

'sp\_upsert\_lkp\_cities',

'sp\_upsert\_lkp\_zips',

'sp\_upsert\_lkp\_location\_names',

'sp\_upsert\_lkp\_retailers',

'sp\_upsert\_lkp\_statuses',

'sp\_upsert\_lkp\_departments',

'sp\_upsert\_lkp\_employees',

'sp\_upsert\_lkp\_customers',

'sp\_upsert\_lkp\_sales',

'p\_load\_ce\_game\_types',

'p\_load\_ce\_game\_categories',

'p\_load\_ce\_game\_numbers',

'p\_load\_ce\_payment\_methods',

'p\_load\_ce\_states',

'p\_load\_ce\_cities',

'p\_load\_ce\_zip',

'p\_load\_ce\_location\_names',

'p\_load\_retailer\_license\_numbers',

'p\_load\_ce\_statuses',

'p\_load\_ce\_departments',

'p\_load\_ce\_employees',

'p\_load\_ce\_customers\_scd',

'p\_load\_ce\_sales',

'sp\_upsert\_dim\_game\_numbers',

'sp\_upsert\_dim\_payment\_methods',

'sp\_upsert\_dim\_retailer\_license\_numbers',

'sp\_upsert\_dim\_employees',

'sp\_upsert\_dim\_customers\_scd',

'sp\_insert\_fct\_sales'

)

**order** **by** procedure\_name, log\_id ;

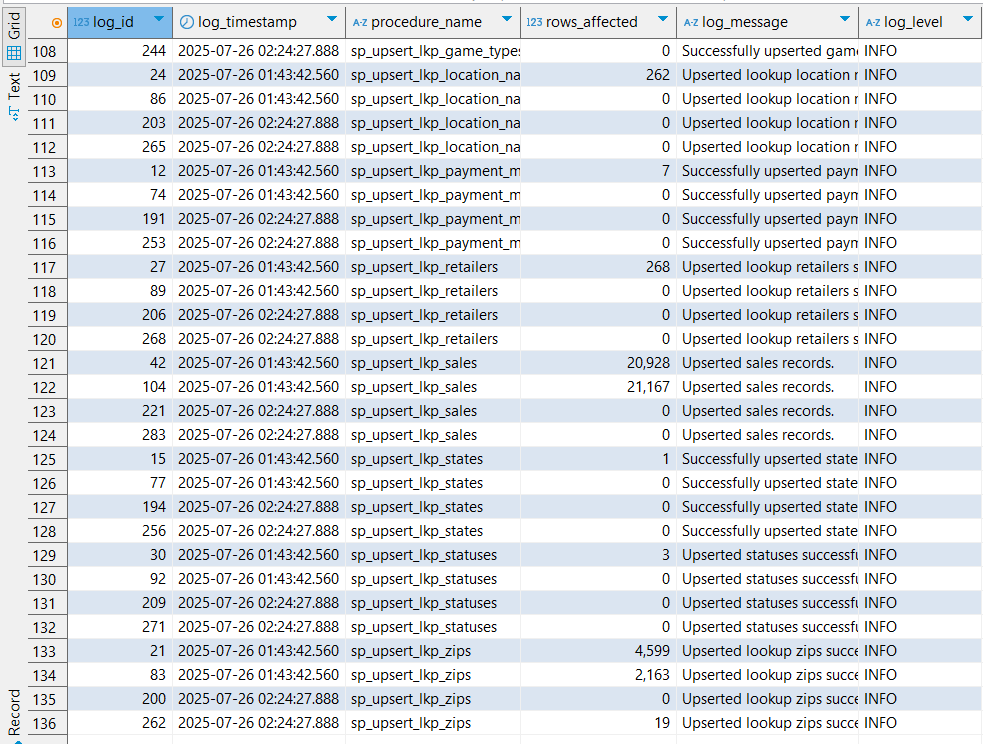
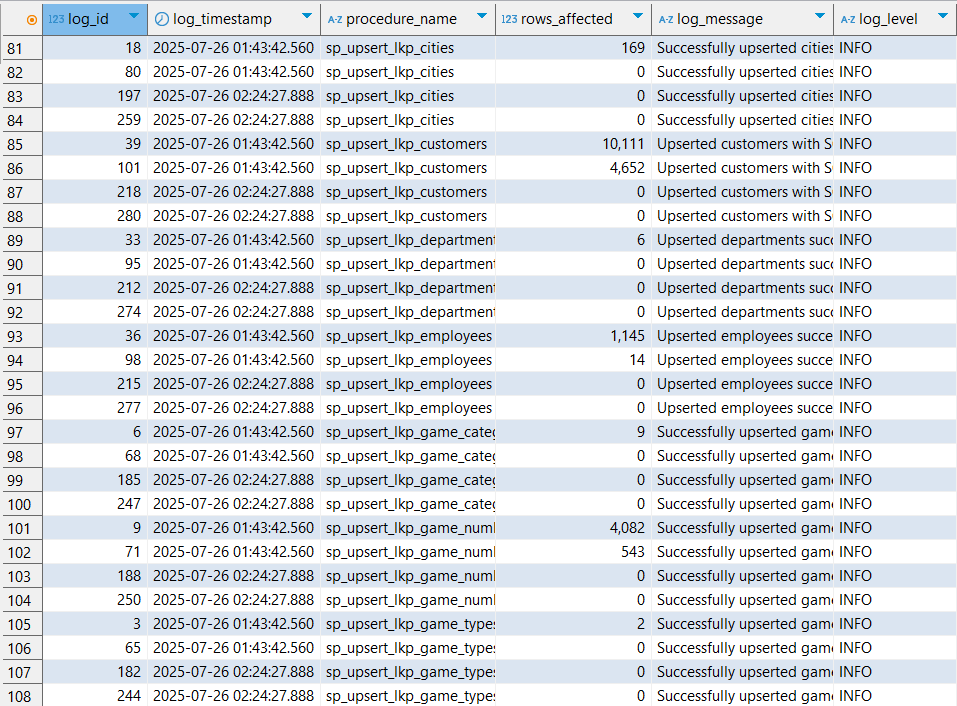
----------------------------------------------------------------------------------------------------------------------------------------

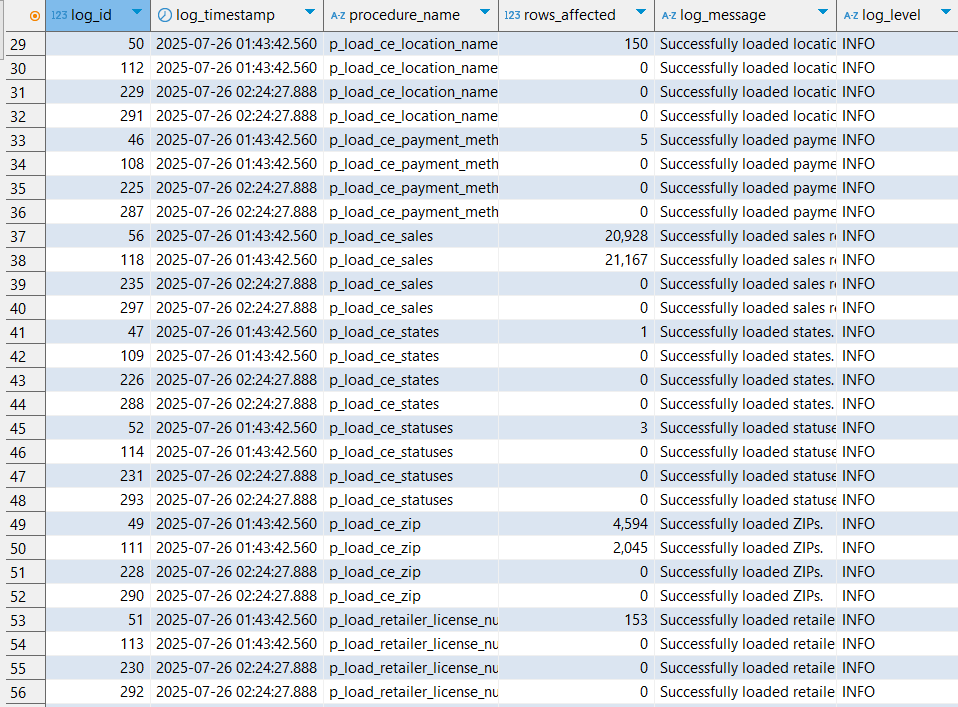
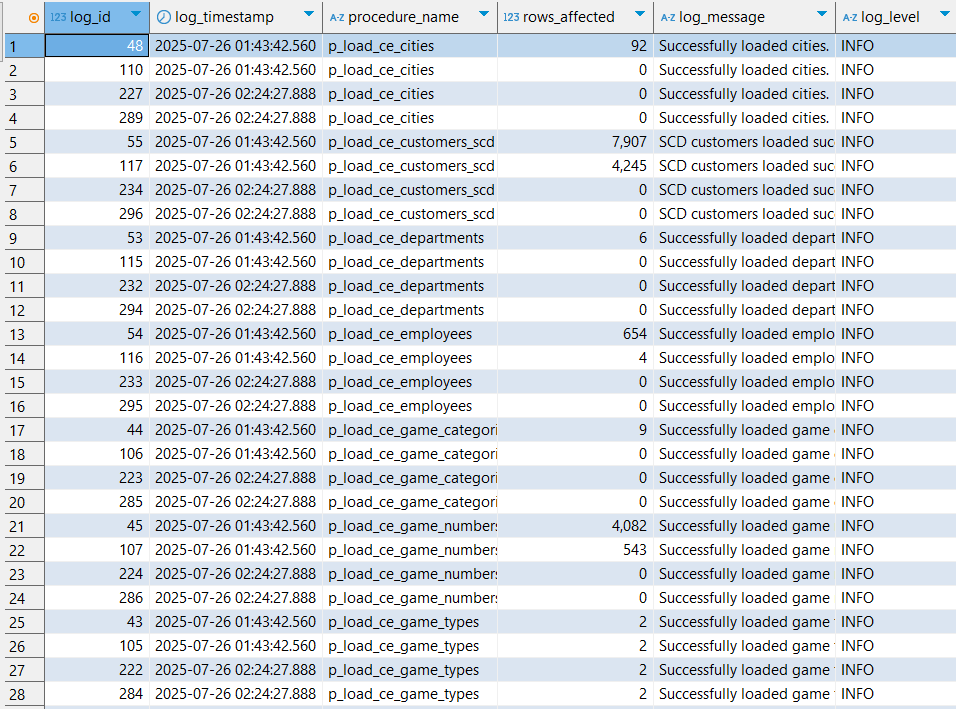
---1.6. Confirm number of rows affected is 0 (or explain why)

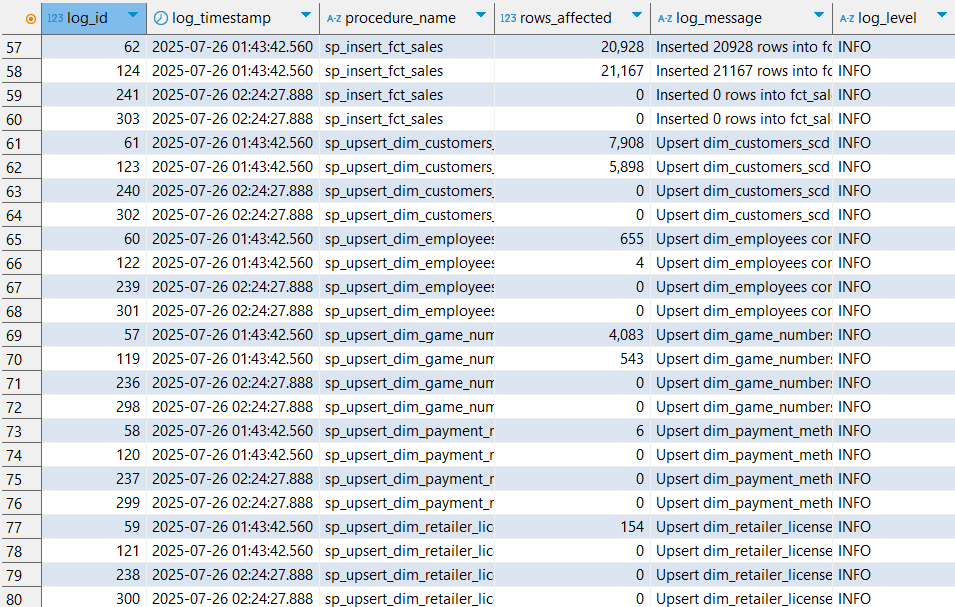
----------------------------------------------------------------------------------------------------------------------------------------

The number of rows is zero because running the process for the same period again will not update any rows based on our pipeline logic. According to this logic, an update only occurs if **at least one of the attributes changes** and the insertion date of the new candidate row is later than the update date of the existing row with the same source ID. Otherwise, the ETL pipeline’s behavior wouldn’t make sense, since we assume row frequency changes happen daily, not within the same day.

Therefore, running the procedure for the same dates again will have no effect—any candidate row will be rejected, and no rows will be updated or inserted.







----Test that the SCD2 procedure works correctly--------------------

--------------------------------------------------------------------------------------------------------------------------

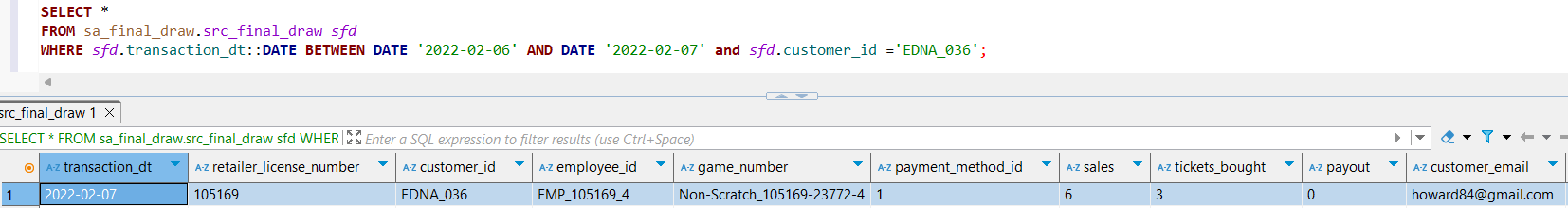
---2.1.Take screenshot of original CSV data (some rows)

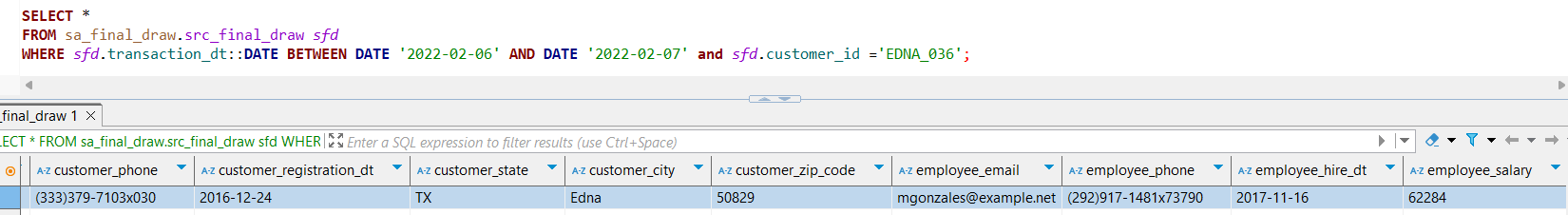
-----------------------------------------------------------------------------------------------

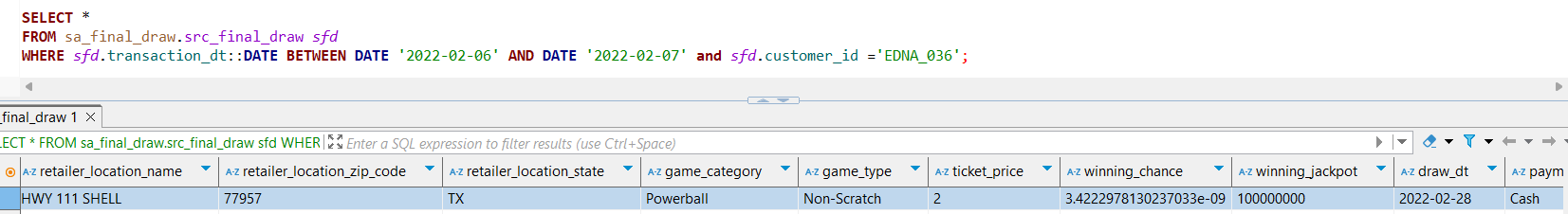
**SELECT** \*

**FROM** sa\_final\_draw.src\_final\_draw *sfd*

**WHERE** *sfd*.transaction\_dt::**DATE** **BETWEEN** **DATE** '2022-02-06' **AND** **DATE** '2022-02-07' **and** *sfd*.customer\_id ='EDNA\_036';



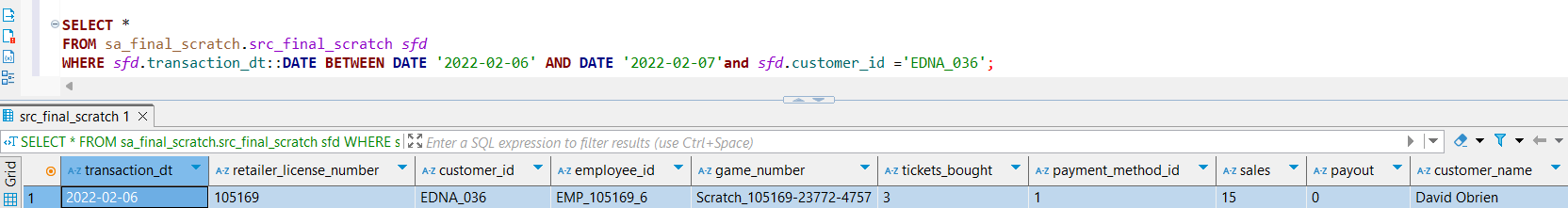


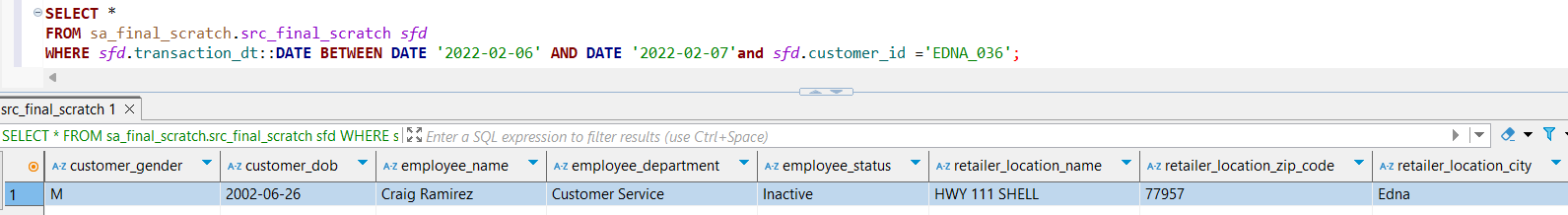


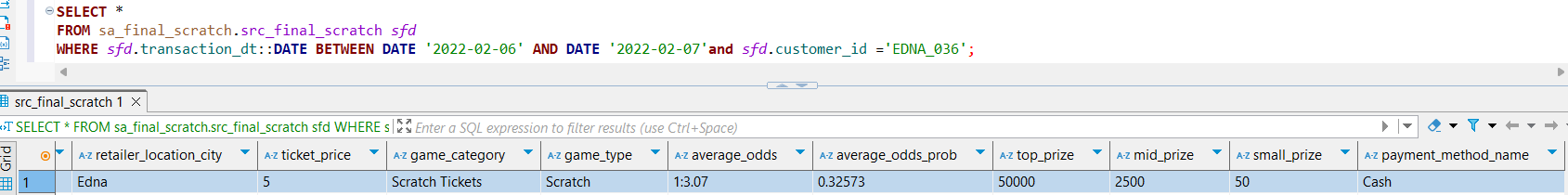
**SELECT** \*

**FROM** sa\_final\_scratch.src\_final\_scratch *sfd*

**WHERE** *sfd*.transaction\_dt::**DATE** **BETWEEN** **DATE** '2022-02-06' **AND** **DATE** '2022-02-07'**and** *sfd*.customer\_id ='EDNA\_036';







**select** \* **from** bl\_cl.lkp\_customers *lc* **where** customer\_src\_id ='EDNA\_036' **order** **by** *lc*.customer\_id **desc**;

**select** \* **from** bl\_3nf.ce\_customers\_scd *ccs* **where** customer\_src\_id ='EDNA\_036' **order** **by** *ccs*.customer\_id **desc** ;

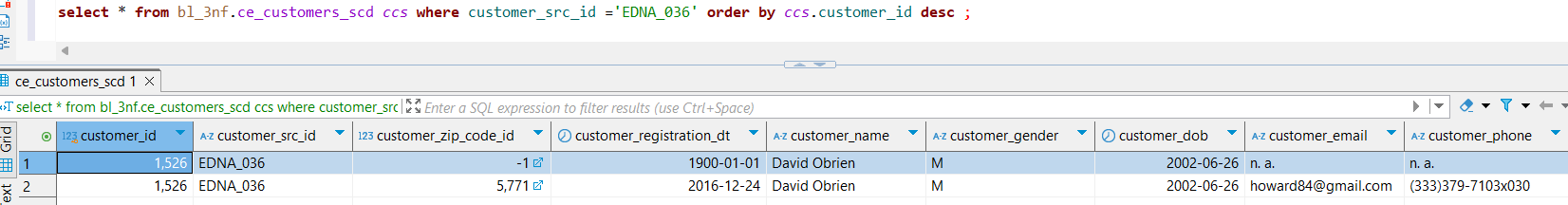
**select** \* **from** bl\_dm.dim\_customers\_scd *dcs* **where** *dcs*.customer\_src\_id ='1526' **order** **by** *dcs*.customer\_src\_id **desc** ;

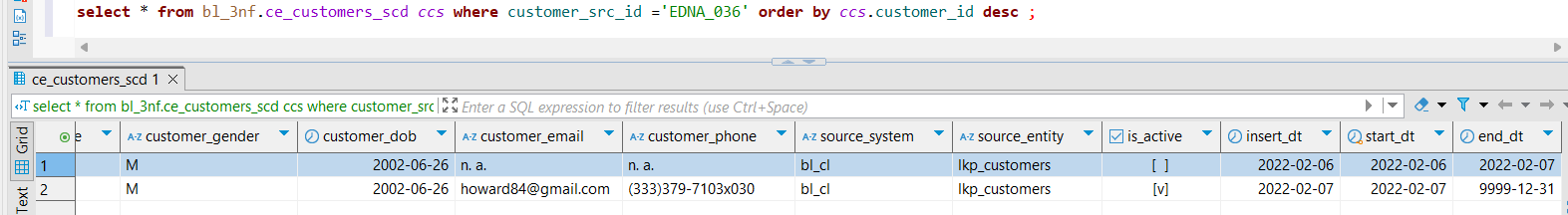
----------------------------------------------------------------------------------------------------------------------------------------

---2.2.Take screenshot of SCD2 data in 3NF and DM layers

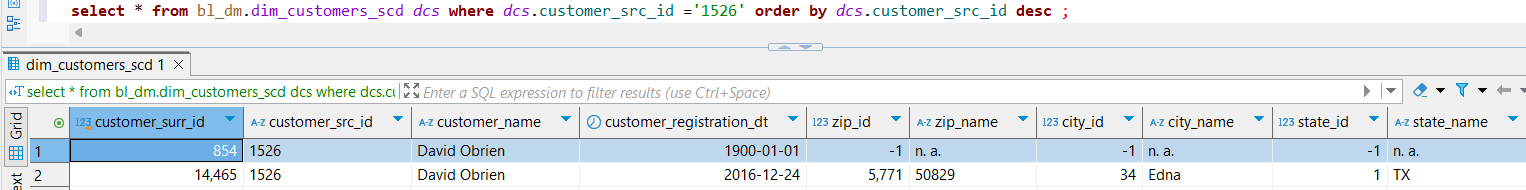
-------------------------------------------------------------------------------------------------------------------------------------------------

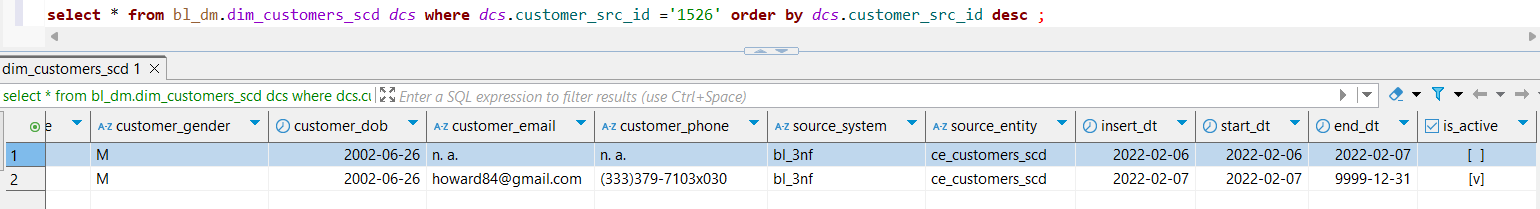
**SELECT** \* **FROM** bl\_3nf.ce\_customers\_scd **WHERE** customer\_src\_id **IN** ('EDNA\_036', 'ANOTHER\_ID') **ORDER** **BY** customer\_id;





**SELECT** \* **FROM** bl\_dm.dim\_customers\_scd **WHERE** customer\_src\_id **IN** ('EDNA\_036', 'ANOTHER\_ID') **ORDER** **BY** customer\_surr\_id;



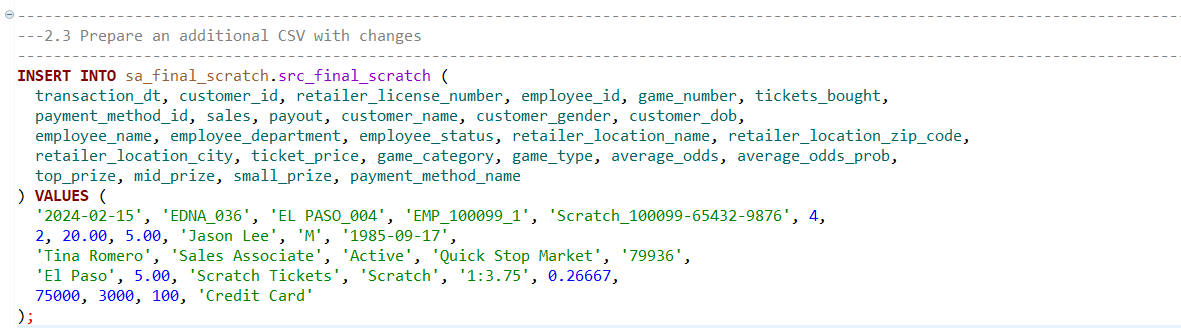


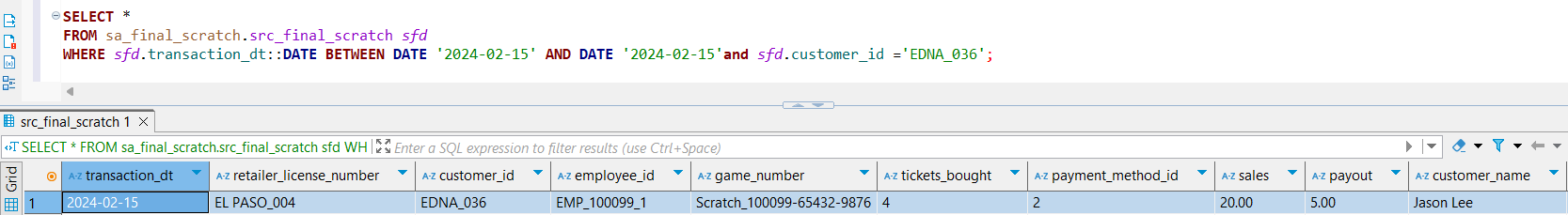
----------------------------------------------------------------------------------------------------------------------------------------

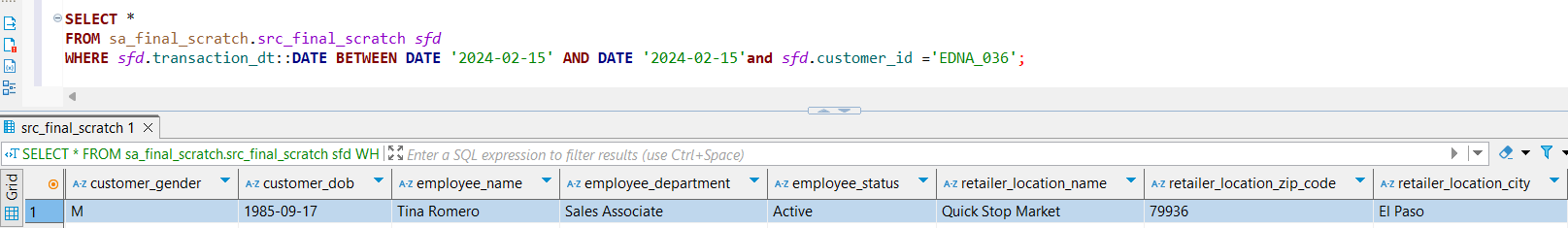
---2.3 Prepare an additional CSV with changes

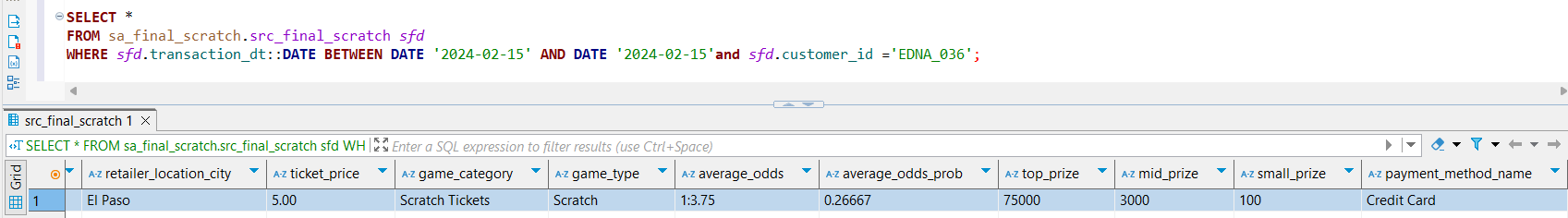
-----------------------------------------------------------------------------------------------------------------------------------------------

I have added a new transaction to the table for the customer with ID **edna\_036**, where the customer’s name and email were updated. This entry was inserted directly into the table. (A separate CSV file containing additional customers will be uploaded to GitHub; however, this example serves solely to demonstrate my SCD Type 2 workaround.)









----------------------------------------------------------------------------------------------------------------------------------------

--- 2.4 Run the loading procedure for the updated CSV

--------------------------------------------------------------------------------------------------------------------------------------------------

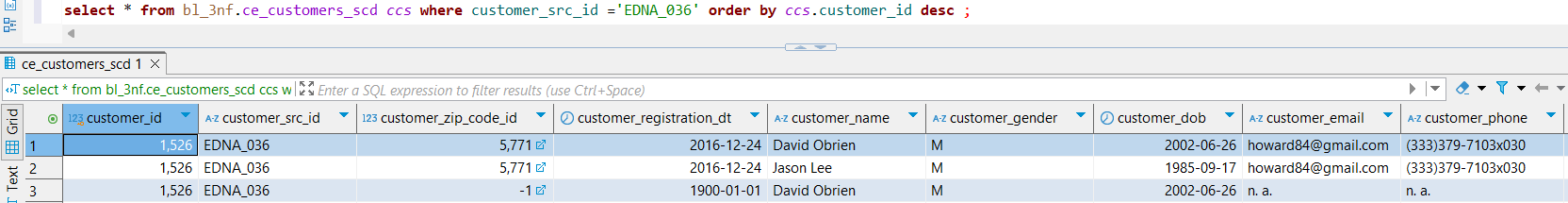
**CALL** bl\_cl.sp\_run\_batch\_etl\_by\_day('2024-02-15', '2024-02-15');

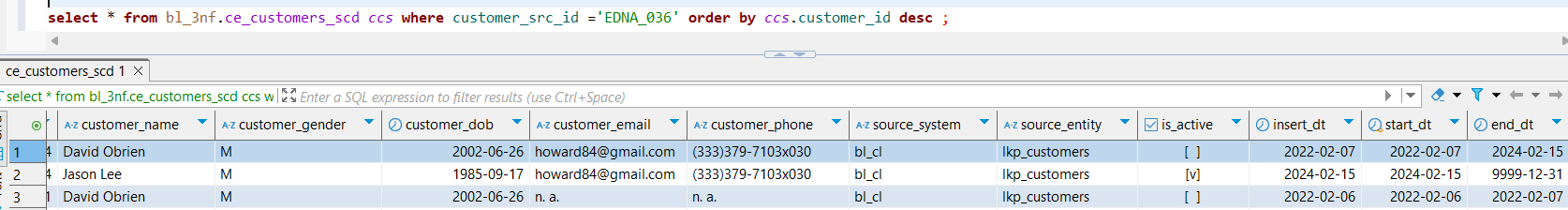
----------------------------------------------------------------------------------------------------------------------------------------

---2.5 Take screenshot showing updated SCD2 data in 3NF and DM layers

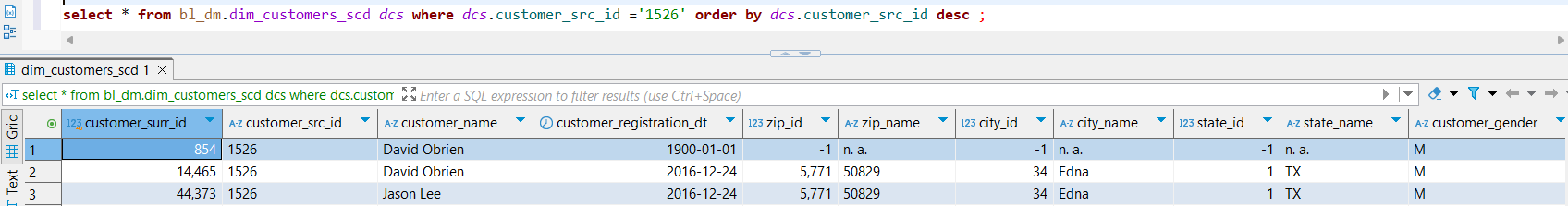
--------------------------------------------------------------------------------------------------------------------------------------------------

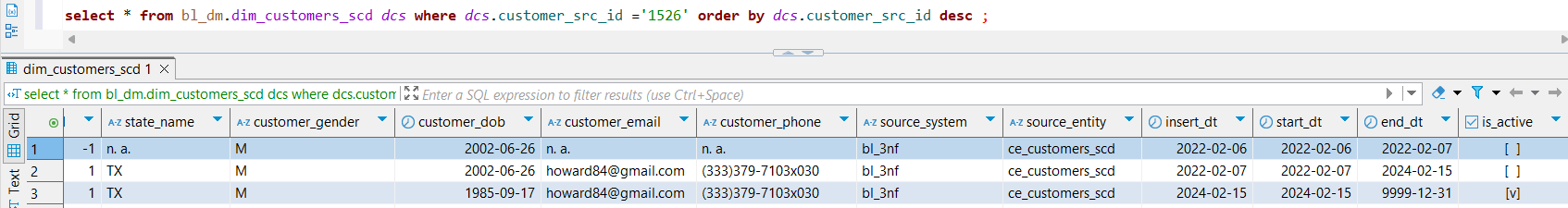
**select \* from bl\_3nf.ce\_customers\_scd *ccs* where customer\_src\_id ='EDNA\_036' order by *ccs*.customer\_id desc ;**

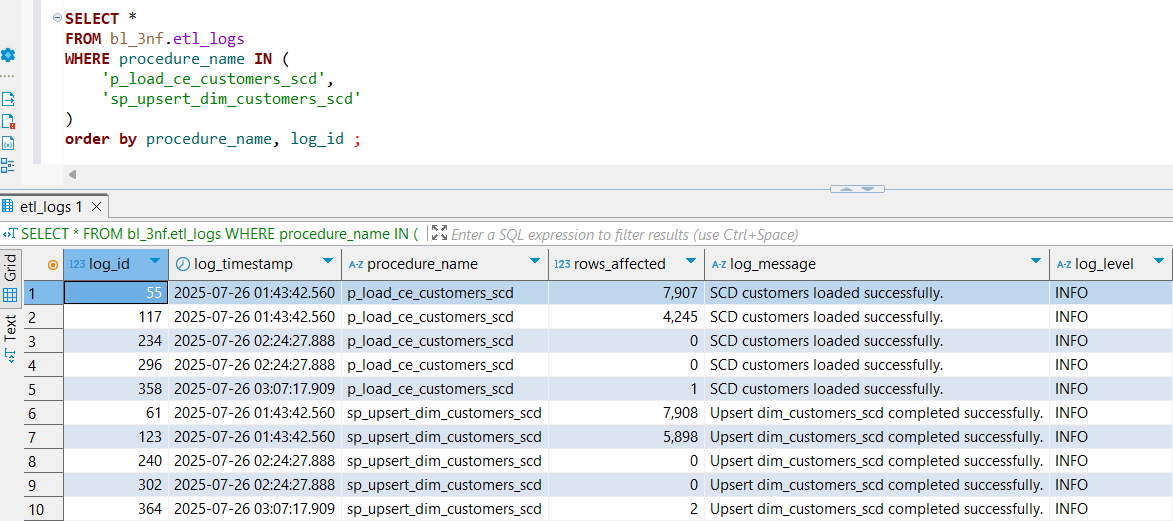
****

****

**select \* from bl\_dm.dim\_customers\_scd *dcs* where *dcs*.customer\_src\_id ='1526' order by *dcs*.customer\_src\_id desc ;**

****

****

****

As shown in the picture, the number of rows affected by this new entry is one higher than in the previous case. This difference arises from how updates are processed in the 3NF model compared to the dimensional model (DM), both of which follow Slowly Changing Dimension (SCD) Type 2 logic but implement it differently.

In the 3NF model, an update is handled as a single atomic operation: the existing row that becomes inactive has its is\_active flag set to false, and the new row with updated attributes is inserted simultaneously. This means the history is preserved by marking the old record inactive and adding a new version for the changes, all in one step.

In the DM approach, this process is split into two operations: first, the new row is inserted as the current active record, and second, the old record is updated to set its is\_active flag to false. This separation leads to seeing a higher number of update operations in the DM pipeline because inserts and updates are counted independently.

Moreover, on the first day, we observe that the count of updated rows is higher by one in both models, even though no additional rows required deactivation. This is due to the insertion of a default row in the 3NF CE table along with the new data rows, which contributes to the total number of affected rows.

When adding a new transaction with new characteristics for a past customer—typical in SCD Type 2 scenarios—both 3NF and DM models correctly apply their logic to capture the change. The 3NF model does this by marking the old row inactive and inserting the new one atomically, while the DM model performs these as two sequential steps. Both approaches ensure historical accuracy and data integrity by maintaining previous states and adding new versions as separate records.