How and Why We Use Control Tables

This text is based on the PoC for ingestions of batches BND_ADJD_MCE into adjd_mce_diag

Control tables are used to track processed data batches. In our case, we have two main control tables: 1. UDW Control Table (source): `dev_osprai.default.btch_cyc` 2. DBR Control Table (target): `dev_osprai.ea_raw.ocs_magnum_control_batch`

These tables help us manage incremental data loads for our main data table: 3. Claim Diagnosis Table (target): `dev_osprai.ea_raw.adjd_mce_diag`

Sequence of Events with Transformations:

1. Check the UDW Control Table (`dev_osprai.default.btch_cyc`) for new batches of data.
2. Compare with the DBR Control Table (`dev_osprai.ea_raw.ocs_magnum_control_batch`) to identify new batches.
3. Process only the new batches of data: a. Fetch new data into `claim_diagnosis_df` DataFrame. b. Apply transformations to `claim_diagnosis_df`:
    - Add 'de_imported_date_ts' column with current timestamp.
    - Cast 'INSRT_BTCH_ID' to long integer.
    - Rename 'INSRT_BTCH_ID' to 'BTCH_ID'.
4. Update the Claim Diagnosis Table (`dev_osprai.ea_raw.adjd_mce_diag`) with the transformed data.
5. Update the DBR Control Table with information about the newly processed batches.

Worked Example:

Initial State: - UDW Control Table (`dev_osprai.default.btch_cyc`): `BTCH_ID | BTCH_NM | STRT_TMSTMP | BTCH_STS | END_TM_STMP 7762927 | BND_ADJD_MCE| 10/10/2023 20:30 | COMPLETE | 11/10/2023 05:22 7763012 | BND_ADJD_MCE| 09/10/2023 20:31 | COMPLETE | 09/10/2023 22:11 7758933 | BND_ADJD_MCE| 08/10/2023 12:50 | COMPLETE | 08/10/2023 21:37 ... | ... | ... | ... | ... (11 rows in total)` - DBR Control Table (`dev_osprai.ea_raw.ocs_magnum_control_batch`): `empty` - Claim Diagnosis Table (`dev_osprai.ea_raw.adjd_mce_diag`): `empty`

Step 1: Check for new batches - We compare `dev_osprai.default.btch_cyc` (11 rows) with `dev_osprai.ea_raw.ocs_magnum_control_batch` (0 rows). - We find that all 11 batches are new.

Step 2: Process new batches - We use our SQL query to ingest data for all 11 batches into `claim_diagnosis_df`: `INSRT_BTCH_ID | UDW_MED_CLM_ID | DIAG_CD | PROC_DT | ... (other columns) 7762927 | ... | ... | ...`

`| ... 7763012 | ... | ... | ... | ... ... | ... | ... | ... | ... (33 rows in total)`

- Apply transformations to `claim_diagnosis_df`: `BTCH_ID | UDW_MED_CLM_ID | DIAG_CD | PROC_DT | ... | de_imported_date_ts 7762927 | ... | ... | ... | ... | 2024-09-27 14:33:32 7763012 | ... | ... | ... | ... | 2024-09-27 14:33:32 ... | ... | ... | ... | ... | ... (33 rows in total)`

Step 3: Update Claim Diagnosis Table - We write the transformed data (33 rows) to `dev_osprai.ea_raw.adjd_mce_diag`.

Step 4: Update DBR Control Table - We add entries for the processed batches to `dev_osprai.ea_raw.ocs_magnum_control_batch`: `ID | BTCH_ID | BTCH_NM | TB_NAME | UDW_STRT_TMSTMP | UDW_BTCH_STS | UDW_END_TM_STMP | OCS_STRT_TMSTMP | OCS_BTCH_STS | OCS_END_TM_STMP 0230ebb44e184105e0c2837cd580428d | 7762927 | BND_ADJD_MCE| raw_udw.ADJD_MCE| 10/10/2023 20:30 | COMPLETE | 11/10/2023 05:22 | 2024-09-27 14:33:39.899| COMPLETE | null 0b0c38942a9fa9e0e16b54675ddc1e0d | 7763012 | BND_ADJD_MCE| raw_udw.ADJD_MCE| 09/10/2023 20:31 | COMPLETE | 09/10/2023 22:11 | 2024-09-27 14:33:39.899| COMPLETE | null ba54c55cc5d27214de35d9a2fea08189 | 7758933 | BND_ADJD_MCE| raw_udw.ADJD_MCE| 08/10/2023 12:50 | COMPLETE | 08/10/2023 21:37 | 2024-09-27 14:33:39.899| COMPLETE | null ... | ... | ... | ... | ... | ... | ... | ... | ... | ... (11 rows in total)`

Next Run (With New Batch IDs):

Assume that after our initial run, new data was loaded into UDW, resulting in new entries in the `btch_cyc` table. Here's how the next run would look:

1. Initial State: - UDW Control Table (`dev_osprai.default.btch_cyc`): `BTCH_ID | BTCH_NM | STRT_TMSTMP | BTCH_STS | END_TM_STMP 7762927 | BND_ADJD_MCE| 10/10/2023 20:30 | COMPLETE | 11/10/2023 05:22 ... | ... | ... | ... | ... 7764001 | BND_ADJD_MCE| 11/10/2023 20:30 | COMPLETE | 12/10/2023 05:22 7764002 | BND_ADJD_MCE| 12/10/2023 20:31 | COMPLETE | 13/10/2023 22:11 (13 rows in total - 11 original + 2 new)`

- DBR Control Table (`dev_osprai.ea_raw.ocs_magnum_control_batch`): `BTCH_ID | BTCH_NM | ... (other columns as before) 7762927 | BND_ADJD_MCE| ... ... | ... | ... 7751699 | BND_ADJD_MCE| ... (11 rows from previous run)`

1. Check for new batches: - Compare `dev_osprai.default.btch_cyc` (13 rows) with `dev_osprai.ea_raw.ocs_magnum_control_batch` (11 rows). - Identify 2 new batches: 7764001 and 7764002.

2. Process new batches: - Ingest data for batches 7764001 and 7764002 into `claim_diagnosis_df`: `INSRT_BTCH_ID | UDW_MED_CLM_ID | DIAG_CD | PROC_DT | ... (other columns) 7764001 | ... | ... |`

`... | ... 7764002 | ... | ... | ... | ... (Let's assume 6 new rows in total)`

- Apply transformations to `claim_diagnosis_df`: `BTCH_ID | UDW_MED_CLM_ID | DIAG_CD | PROC_DT | ... | de_imported_date_ts 7764001 | ... | ... | ... | ... | 2024-09-28 10:15:32 7764002 | ... | ... | ... | ... | 2024-09-28 10:15:32 ... | ... | ... | ... | ... | ... (6 rows with transformations applied)`

1. Update Claim Diagnosis Table: - Write the transformed data (6 new rows) to `dev_osprai.ea_raw.adjd_mce_diag`. - The table now contains 39 rows in total (33 from previous run + 6 new).

2. Update DBR Control Table: - Add entries for the newly processed batches to `dev_osprai.ea_raw.ocs_magnum_control_batch`: `ID | BTCH_ID | BTCH_NM | TB_NAME | UDW_STRT_TMSTMP | UDW_BTCH_STS | UDW_END_TM_STMP | OCS_STRT_TMSTMP | OCS_BTCH_STS | OCS_END_TM_STMP (existing 11 rows...) f123abc456def789ghi0123jkl456mn | 7764001 | BND_ADJD_MCE| raw_udw.ADJD_MCE| 11/10/2023 20:30 | COMPLETE | 12/10/2023 05:22 | 2024-09-28 10:15:39.123| COMPLETE | null a987cba654fed321ihg9876lkj321po | 7764002 | BND_ADJD_MCE| raw_udw.ADJD_MCE| 12/10/2023 20:31 | COMPLETE | 13/10/2023 22:11 | 2024-09-28 10:15:39.123| COMPLETE | null (13 rows in total - 11 existing + 2 new)`

3. Final State: - UDW Control Table: 13 rows (unchanged) - DBR Control Table: 13 rows (updated with 2 new entries) - Claim Diagnosis Table: 39 rows (33 existing + 6 new)

This example demonstrates how the process: 1. Identifies only the new batch IDs (7764001 and 7764002) by comparing the UDW and DBR control tables. 2. Processes only the data for these new batch IDs. 3. Updates the Claim Diagnosis Table with only the new data. 4. Adds entries for the newly processed batches to the DBR Control Table.

This incremental approach ensures that: - Only new data is processed in each run. - The process is efficient, avoiding reprocessing of already ingested data. - The control tables accurately reflect the current state of data ingestion.