Extracting From a DBMS: Incremental

School of Information Studies
Syracuse University

Incremental CET LSET

- You extract only what you have not extracted previously.
- CET →Current extraction timestamp.
- LSET → Last successful extraction timestamp.
- CET and LSET are stored in a metadata table.
- Fault tolerant. If it fails, it can be rerun to pick up data it missed.
- When extraction is complete LSET is set to CET.
- When to use this approach:
  - For any OLTP sources that include metadata columns indicating when the row was created or last updated. Cannot be used otherwise.

Example: CET and LSET

OLTP source customers

Customer ID	Customer Name	Customer Credit	Created On	Last Update On
1001	Robin Banks	$4000	3/2/2015	7/11/2016
1002	Jean Poole	$1500	5/25/2016	7/12/2016
1003	Max Emum	$3200	7/13/2016	7/13/2016

Metadata: incr_extract

Extract ID	Source	Table Name	LSET	CET
1	fudgemart	customers	7/11/2016	7/13/2016

```
SELECT * FROM fudgemart.customers
WHERE [Created On] > LSET and [Created On] <= CET
AND [Last Update On] > LSET and [Last Update On] <= CET
```

Which row(s) will be extracted? Which row(s) were extracted already?

Incremental Other Source Column

- Use a PK, date column, or business key of the source system for incremental loads.
- Metadata table used to keep track of current extraction ID (CEID) and last successful extraction ID (LSEID).
- Fault tolerant.
- When extraction is complete, LSEID is set to CET.
- When to use this approach:
  - Works for transaction fact tables; cannot track updates to dimensions or master data this way.

Example: Incremental Other

OLTP source customers

Customer ID	Customer Name	Customer Credit
1001	Robin Banks	$4000
1002	Jean Poole	$1500
1003	Max Emum	$3200

Metadata: incr_extract

Extract ID	Source	Table Name	LSEID	CEID
1	fudgemart	customers	1001	1003

```
SELECT * FROM fudgemart.customers
WHERE [Customer ID] > LSET and [Customer ID] <= CET
```

Which row(s) will be extracted? Which row(s) were extracted already?

Fixed Range

- Useful for very large source tables that take a very long time to query.
- Extract data in batches by year or month, for example.
- A good strategy for one-time extracts such as transactions and completed accumulating snapshots with millions of rows.