Параметр INSERTMISSINGUPDATES в Oracle GoldenGate: как работает и где может пригодиться

Параметр **`INSERTMISSINGUPDATES`** **(`NOINSERTMISSINGUPDATES`)** управляет поведением процесса **Replicat** в ситуации, когда для операции обновления (UPDATE) не найдена целевая запись. Если параметр включён, **Replicat** вставит новую строку на основе данных из источника; если выключен (значение по умолчанию) – запись отсутствует, возникает ошибка, и дальнейшая судьба транзакции зависит от настройки `REPERROR`.

Как это работает по документации

`INSERTMISSINGUPDATES` следует использовать только тогда, когда исходная база данных гарантированно логирует **все** столбцы строки (даже те, что не изменились). Это необходимо, потому что для вставки новой записи требуются значения всех столбцов, а не только изменённых.

- Если источник передаёт только изменённые столбцы **(сжатый формат обновлений),** параметр всё же может работать при условии, что целевая таблица допускает `NULL` в отсутствующих колонках.
- Если же база данных по умолчанию логирует все столбцы (например, включено дополненное логирование), то для корректной работы **`INSERTMISSINGUPDATES`** необходимо также использовать параметры **`NOCOMPRESSUPDATES`** и **`NOCOMPRESSDELETES`** (если они поддерживаются). В противном случае потребуется **`FETCHOPTIONS MISSINGCOLS`**, чтобы добрать недостающие колонки из источника.

Важно помнить: параметр является таблично-специфичным и действует для всех последующих операторов `MAP`, пока не встретится противоположная директива.

Дополнительные требования к источнику

Из описания ясно, что для надёжной работы **INSERTMISSINGUPDATES** на таблицах источника должно быть включено дополненное логирование всех столбцов:

```
ALTER TABLE SCHEME.CUSTOMERS ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

Без этого при обновлении в трейл попадут только изменившиеся колонки, и **Replicat** не сможет сформировать полноценную вставку.

Неочевидный сценарий использования: заполнение пробелов

Параметр может пригодиться не только для обработки редких ситуаций рассинхронизации, но и для целенаправленного «долива» пропущенных данных. Предположим, в таблице SCHEME.CUSTOMERS по какой-то причине образовались пробелы (часть строк отсутствует на приёмнике), мы знаем, что DELETE-операций в этом наборе нет. Мы знаем временной интервал, в котором могли быть пропуски.

Тогда можно выполнить на источнике фиктивный UPDATE, который затронет все потенциально пропущенные строки:

```
UPDATE SCHEME.CUSTOMERS SET ID = ID
WHERE DATE_CHANGE BETWEEN TO_DATE('...') AND TO_DATE('...');
```

Поскольку условие ID = ID не меняет данные, на источнике не происходит реальных изменений, но в трейл всё равно попадут образы строк (**NOCOMPRESSUPDATES** за это отвечает). На приёмнике, если строка уже существует, обновление пройдёт без эффекта; если же строки нет, **INSERTMISSINGUPDATES** вставит её. Так можно аккуратно засинкать таблицы, без полной перезаливки или использования Веридаты.

Грабли: порядок операций

Несмотря на заверения оракл, что **Replicat** применяет изменения в том же порядке, что и на источнике, на практике возможны спецэффекты. Например, если после всех обновлений строки она была удалена, но из-за особенностей транспорта или параллельной работы DELETE прилетит раньше финального UPDATE, то на приёмнике картина может исказиться:

Источник	Приёмник (ожидание)	Реальность при нарушении порядка
INSERT	INSERT	INSERT
UPDATE	UPDATE	UPDATE
UPDATE	UPDATE	UPDATE
UPDATE	UPDATE	DELETE (пришёл раньше последнего UPDATE)
DELETE	DELETE	UPDATE (вставляет пропущенную строку!)

В результате удалённая на источнике строка снова появится на приёмнике из-за **`INSERTMISSINGUPDATES`**, который вставит её при обработке запоздавшего UPDATE. Поэтому включать параметр'чтоб было' - не стоит.

Revision #2
Created 24 February 2026 07:25:43 by catmaster
Updated 25 February 2026 10:25:04 by catmaster