Create new index for performance issue

Process - Receive Shop order with lot batches.

A bug correction added new check when receiving shop order parts.

Inventory_Transaction_Hist_API.Check_Lot_Batch_In_Use(part_no_, lot_batch_no_);

This is called from reserved_lot_batch_api.is_reservation_allowed when receiving shop orders. This procedure is very slow on our system as **inventory_transaction_hist_tab** contain **8.5 million** rows on our system.

Trace files shows following information for receive process.

SELECT 1

FROM

INVENTORY_TRANSACTION_HIST_TAB T WHERE PART_NO = :B2 AND LOT_BATCH_NO = :B1

AND DIRECTION IN ('+', '-') AND (QUANTITY - QTY_REVERSED) != 0

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	1	0.00	0.00	0	0	0	0
Fetch	1	1.15	90.03	13240	13265	0	0
total	3	1.15	90.04	13240	13265	0	0

Misses in library cache during parse: 1

Misses in library cache during execute: 1

Optimizer mode: ALL_ROWS

Parsing user id: 81 (IFSAPP) (recursive depth: 1)

Number of plan statistics captured: 1

Rows (1st) Rows (avg) Rows (max) Row Source Operation

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

0 0 0 TABLE ACCESS BY INDEX ROWID BATCHED INVENTORY_TRANSACTION_HIST_TAB (cr=13265 pr=13240 pw=0 time=90038741 us cost=942 size=32 card=1)

24747 24747 24747 INDEX RANGE SCAN INVENTORY_TRANSACTION_HIST_IX1 (cr=269 pr=268 pw=0 time=897320 us cost=19 size=0 card=20989)(object id 101810)

Rows Execution Plan

------- --

0 SELECT STATEMENT MODE: ALL_ROWS

0 TABLE ACCESS MODE: ANALYZED (BY INDEX ROWID BATCHED) OF

'INVENTORY_TRANSACTION_HIST_TAB' (TABLE)

24747 INDEX MODE: ANALYZED (RANGE SCAN) OF

'INVENTORY_TRANSACTION_HIST_IX1' (INDEX)

According to the trace file report elapsed time is high for fetch which having lot of disk reads. However, query goes through the INVENTORY_TRANSACTION_HIST_IX1 index but performance was bad.

INVENTORY_TRANSACTION_HIST_IX1 = `INVENTORY_TRANSACTION_HIST_TAB (PART_NO, CONTRACT, DATE_APPLIED)`

Since index have only part_no as a predicate, existing index doesn't help much to improve performance.

Solution is to create new index using part_no and lot_batch_no. it improves the performance.