

- /*+SWAP_JOIN_INPUTS */: 해시조인의 경우, BUILD INPUT를 명시적으로 선택

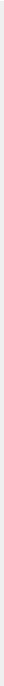
EX: /*+ SWAP_JOIN_INPUTS(A)*/

--해시조인의경우 BUILD INPUT과 PROBE에 대한 순서를 정할 수 있다.

첫 번째 알고리즘

```
select /*+ leading(r, c, l, d, e)
      use_hash(c) use_hash(l) use_hash(d) use_hash(e) */
      e.first_name, e.last_name, d.department_name
      , l.street_address, l.city, c.country_name, r.region_name
from    hr.regions r
      , hr.countries c
      , hr.locations l
      , hr.departments d
      , hr.employees e
where    d.department_id = e.department_id
and      l.location_id = d.location_id
and      c.country_id = l.country_id
and      r.region_id = c.region_id;
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		106	10706	15 (14)	00:00:01
* 1	HASH JOIN		106	10706	15 (14)	00:00:01
* 2	HASH JOIN		27	2241	12 (17)	00:00:01
* 3	HASH JOIN		23	1472	8 (13)	00:00:01
* 4	HASH JOIN		25	700	5 (20)	00:00:01
5	TABLE ACCESS FULL	REGIONS	4	56	3 (0)	00:00:01
6	INDEX FULL SCAN	COUNTRY_C_ID_PK	25	350	1 (0)	00:00:01
7	TABLE ACCESS FULL	LOCATIONS	23	828	3 (0)	00:00:01
8	TABLE ACCESS FULL	DEPARTMENTS	27	513	3 (0)	00:00:01
9	TABLE ACCESS FULL	EMPLOYEES	107	1926	3 (0)	00:00:01



두 번째 알고리즘

```
select /*+ leading(r, c, l, d, e)
        use_hash(c) use_hash(l) use_hash(d) use_hash(e)
        swap_join_inputs(l)
        swap_join_inputs(d)
        swap_join_inputs(e) */
       e.first_name, e.last_name, d.department_name
,      l.street_address, l.city, c.country_name, r.region_name
from    hr.regions r
,      hr.countries c
,      hr.locations l
,      hr.departments d
,      hr.employees e
where   d.department_id = e.department_id
and     l.location_id = d.location_id
and     c.country_id = l.country_id
and     r.region_id = c.region_id;
```

Id	Operation	Name	Rows	Bytes	Cost	(%CPU)	Time
0	SELECT STATEMENT		106	10706	15	(14)	00:00:01
* 1	HASH JOIN		106	10706	15	(14)	00:00:01
2	TABLE ACCESS FULL	EMPLOYEES	107	1926	3	(0)	00:00:01
* 3	HASH JOIN		27	2241	12	(17)	00:00:01
4	TABLE ACCESS FULL	DEPARTMENTS	27	513	3	(0)	00:00:01
* 5	HASH JOIN		23	1472	8	(13)	00:00:01
6	TABLE ACCESS FULL	LOCATIONS	23	828	3	(0)	00:00:01
* 7	HASH JOIN		25	700	5	(20)	00:00:01
8	TABLE ACCESS FULL	REGIONS	4	56	3	(0)	00:00:01
9	INDEX FULL SCAN	COUNTRY_C_ID_PK	25	350	1	(0)	00:00:01

```
select /*+ leading(d, e, l, c, r)
        use_hash(e) use_hash(l) use_hash(c) use_hash(r)
        swap_join_inputs(l)
        swap_join_inputs(c)
        swap_join_inputs(r) */
       e.first_name, e.last_name, d.department_name
,      l.street_address, l.city, c.country_name, r.region_name
from    hr.regions r
,      hr.countries c
,      hr.locations l
,      hr.departments d
,      hr.employees e
where   d.department_id = e.department_id
and     l.location_id = d.location_id
and     c.country_id = l.country_id
and     r.region_id = c.region_id;
```

Id	Operation	Name	Rows	Bytes	Cost	(%CPU)	Time
0	SELECT STATEMENT		106	10706	15	(14)	00:00:01
* 1	HASH JOIN		106	10706	15	(14)	00:00:01
2	TABLE ACCESS FULL	REGIONS	4	56	3	(0)	00:00:01
* 3	HASH JOIN		106	9222	12	(17)	00:00:01
4	INDEX FULL SCAN	COUNTRY_C_ID_PK	25	350	1	(0)	00:00:01
* 5	HASH JOIN		106	7738	10	(10)	00:00:01
6	TABLE ACCESS FULL	LOCATIONS	23	828	3	(0)	00:00:01
* 7	HASH JOIN		106	3922	7	(15)	00:00:01
8	TABLE ACCESS FULL	DEPARTMENTS	27	513	3	(0)	00:00:01
9	TABLE ACCESS FULL	EMPLOYEES	107	1926	3	(0)	00:00:01