**Solutions**

**Task 1:**

select a.id\_class,

c.parameter as parameter\_class,

id\_subclass,

query2.subclassparam as parameter\_subclass

from A as a

join C as c

on a.id\_class = c.id

join (select c.id, c.parameter as subclassparam from A as a join C as c on a.id\_subclass = c.id) as query2

on query2.id=a.id\_subclass;

**Task 2:**

select a.id, coalesce(b.attr, b1.attr) as attr

from "TableA" a

left join "TableB" b on a.cat = b.cat and a.type = b.type

left join "TableB" b1 on a.cat = b1.cat and b1.type='NA'

order by a.id

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**Task #1.** Table A contains IDs of classes and their subclasses:

|  |  |
| --- | --- |
| **TABLE A** | |
| **ID\_CLASS** | **ID\_SUBCLASS** |
| 1 | 1.1 |
| 1 | 1.2 |
| 2 | 2.1 |

Table B contains parameters for all IDs, both classes and subclasses:

|  |  |
| --- | --- |
| **TABLE B** | |
| **ID** | **PARAMETER** |
| 1 | A |
| 2 | B |
| 1.1 | C |
| 1.2 | D |
| 2.1 | E |

Write a SQL query which returns parameters for each class and subclass in such view:

|  |  |  |  |
| --- | --- | --- | --- |
| **RESULT TABLE** | | | |
| **ID\_CLASS** | **PARAMETER\_CLASS** | **ID\_SUBCLASS** | **PARAMETER\_SUBCLASS** |
| 1 | A | 1.1 | C |
| 1 | A | 1.2 | D |
| 2 | B | 2.1 | E |

**Task #2.** Table A contains dynamic data – ID, category and type:

|  |  |  |
| --- | --- | --- |
| **TABLE A** | | |
| **ID** | **CAT** | **TYPE** |
| 1 | ETD | H49A |
| 2 | ETD | SZ8A |
| 3 | ETD | VQA5 |
| 4 | ETD | (null) |
| …. |  |  |
| 16 | OTC | BVX9A |
| …. |  |  |

Static table B contains mapping of category and typecombinations to attributes:

|  |  |  |
| --- | --- | --- |
| **TABLE B** | | |
| **CAT** | **TYPE** | **ATTR** |
| ETD | H49A | W |
| ETD | SZ8A | X |
| ETD | ‘NA’ | Y |
| OTC | BVX9A | Z |
| …. |  |  |

All categories from dynamic Table A are presented in static Table B. But Types from Table A can be missed in Table B.

Attributes are mapped according to the following rules (example for one category):

IF A.CAT = ‘A’ AND A.TYPE = ‘A1’THEN ATTR = ‘1’

ELIF A.CAT = ‘A’ AND A.TYPE = ‘A2’ THEN ATTR = ‘2’

…..

ELIF A.CAT = ‘A’ THEN ATTR = ‘99’

Write a SQL query to get attribute for each ID in such view:

|  |  |
| --- | --- |
| **RESULT TABLE** | |
| **ID** | **ATTR** |
| 1 | W |
| 2 | X |
| 3 | Y |
| 4 | Y |
| … |  |
| 16 | Z |
| … |  |

**Query must be universal and not depend on specific values in table A.**