SQL Functions

1. Consider the table `Exams(sid, cid, score)`, such that:

   - Each `sid` is an integer and represents a student ID.
   - Each `cid` is an integer and represents a course ID.
   - Each `score` is an integer and represents a final exam score of a student in a course.

   Write a function `max_min` with the following properties:

   - It has an input parameter `stu_id`, which is an integer.
   - It has two output parameters `max_cid` and `min_cid`, both of which are integers.
   - It examines the records in `Exams` whose `sid` is equal to `stu_id` and identifies the two records among them with the largest and smallest `score` where *ties are broken arbitrarily*. For the record with the largest `score`, its `cid` is assigned to `max_cid`. For the record with the smallest `score`, its `cid` is assigned to `min_cid` *only if it is smaller than the largest score*. Otherwise, `min_cid` is set to NULL.

   A template for `max_min` is provided below:

```sql
CREATE OR REPLACE FUNCTION max_min (IN stu_id INT, OUT max_cid INT, OUT min_cid
    INT)
RETURNS RECORD AS $$
DECLARE
  max_score INT;
  min_score INT;
BEGIN
  /* write your code here */
END;
$$ LANGUAGE plpgsql;
```

2. Consider the same table `Exams(sid, cid, score)` from the previous question. Write a function `revised_avg` with the following properties:

   - It has an input parameter `stu_id`, which is an integer.
   - It has an output parameter `r_avg`, which is a numeric.
   - It examines the records in `Exams` whose `sid` is equal to `stu_id`. If there are *at least* 3 such records, the function returns the average `score` of these records but *excludes* one record with the highest `score` with *ties broken arbitrarily* as well as one record with the lowest `score` with *ties broken arbitrarily*. If there are fewer than 3 such records, the function returns NULL.

   A template for `revised_avg` is provided below:

```
CREATE OR REPLACE FUNCTION revised_avg (IN stu_id INT, OUT r_avg FLOAT)
RETURNS FLOAT AS $$
  /* write your code here */
$$ LANGUAGE plpgsql;
```

3. Consider the same table `Exams(sid, cid, score)` from the first question as well as the concept of "revised average `score`" in the previous question. Write a function `list_r_avg` that returns the `sid` of each student in `Exams` along with their revised average `score`. For simplicity, we assume that all `sid` in `Exams` are non-negative integers.

   A template for `list_r_avg` is provided below:

```
CREATE OR REPLACE FUNCTION list_r_avg ()
RETURNS TABLE (stu_id INT, ravg FLOAT) AS $$
DECLARE
  curs CURSOR FOR (SELECT sid, score FROM Exams ORDER BY sid);
  /* add other variables here */
BEGIN
  /* write your code here */
END;
$$ LANGUAGE plpgsql;
```

4. Consider the same table `Exams(sid, cid, score)` from the first question. Write a function `list_scnd_highest` that returns the `sid` of each student in `Exams` along with their second highest `score` with *ties broken arbitrarily*. If the student only has one `score`, then the second highest `score` is NULL. For simplicity, we assume that all `sid` in `Exams` are non-negative integers.

   A template for `list_scnd_highest` is provided below:

```
CREATE OR REPLACE FUNCTION list_scnd_highest ()
RETURNS TABLE (stu_id INT, scnd_highest INT) AS $$
  /* write your code here */
$$ LANGUAGE plpgsql;
```