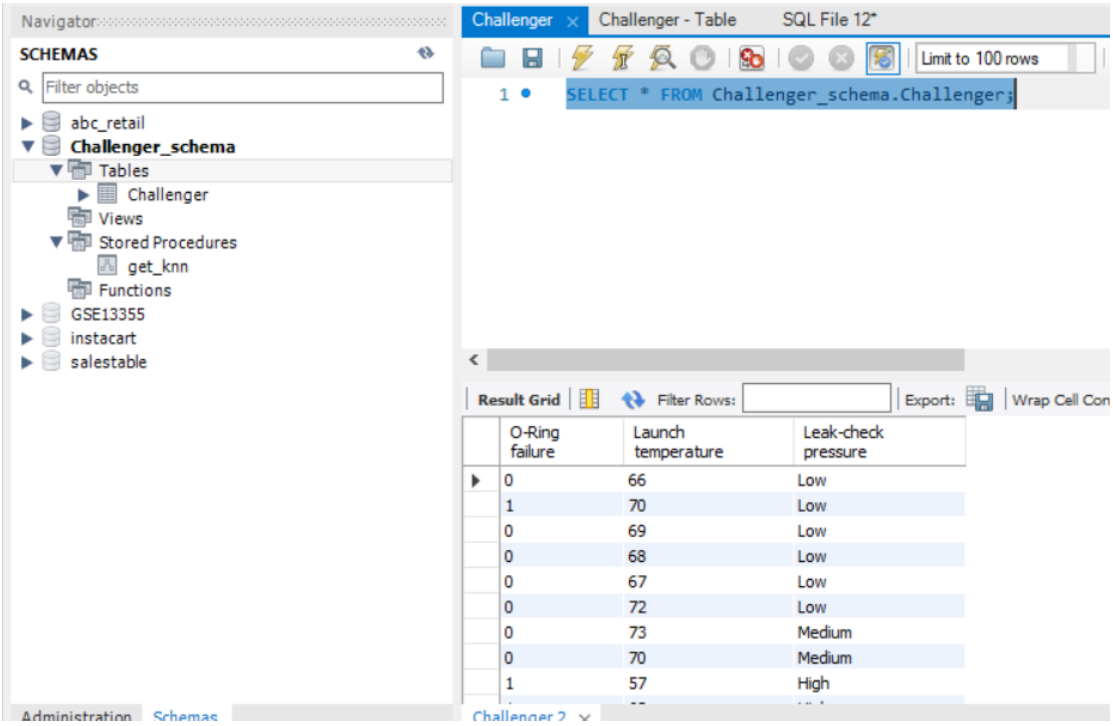
1- Import the attached file (Challenger.csv) to your MySQL.



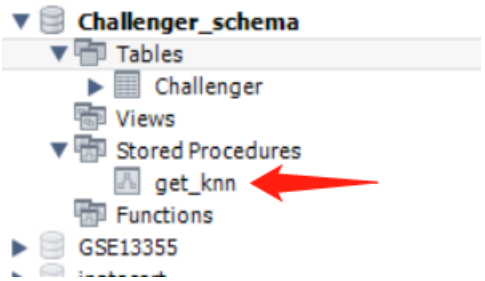
2- Create a stored procedure call KNN with five inputs (@table, @target, @predictor, @value, @k).

```
CREATE DEFINER=`admin`@`%` PROCEDURE `get_knn`(
    in tbl varchar(100), tar varchar(100), predictor varchar(100), val
varchar(10), k int
)
-
- if use col `Launch temperature` as @predictor, we may need to use cas
t to deal with the value
-- elif use col `Leak-
check pressure` we need to convert low/medium/high to int label 1/2/3
--   in order to calculate the Euclidean distance
BEGIN
    -- create temp table for knn data
    -- drop temp table if exists
    DROP TABLE IF EXISTS Challenger_schema.temp1;
    CREATE TABLE Challenger_schema.temp1
    (
        `O-Ring failure` TINYINT(1) NOT NULL,
        `Launch temperature` INT NOT NULL,
        `Leak-check pressure` VARCHAR(10) NOT NULL,
        `ED` DOUBLE NOT NULL
    );

    -- swtich between predictor
    -- calculate knn distance and return a temp view
    IF (predictor='Launch temperature') THEN
        -- @predictor=`Launch temperature`
        -- Find @k nearest neighbors and insert the data
        SET @sql1 = CONCAT(
            'INSERT INTO Challenger_schema.temp1(
                    `O-Ring failure`, `Launch temperature`, `Leak-
check pressure`, `ED`
                )
            SELECT `O-Ring failure`, `Launch temperature`, `Leak-
check pressure`, sqrt(pow((`',
            predictor,
            '` - CAST(',
            val,
            ' as signed)), 2)) AS `ED`
                FROM Challenger_schema.',
```

```
            tbl,
            ' ORDER BY `ED` ASC LIMIT ',
            k,
            ';'
        );
        prepare getsql1 from @sql1;
        execute getsql1;
    ELSEIF (predictor='Leak-check pressure') THEN
        -- Find @k nearest neighbors and insert the data
        SET @sql1 = CONCAT(
            'INSERT INTO Challenger_schema.temp1(
                        `O-Ring failure`, `Launch temperature`, `Leak-
check pressure`, `ED`
                    )
            SELECT `O-Ring failure`, `Launch temperature`, `Leak-
check pressure`, sqrt(pow((CASE
                    WHEN ''',
            val,
            '''=''Low'' THEN `temp_predictor` - 1 WHEN ''',
            val,
            '''=''Medium'' THEN `temp_predictor` - 2 WHEN ''',
            val,
            '''=''High'' THEN `temp_predictor` - 3
                ELSE `temp_predictor` - (-1)
            END
                    ),
                    2
                )) AS `ED`
                    FROM (
                SELECT
                    *,
                    (
                CASE
                    WHEN ``,
            predictor,
            ``=''Low'' THEN 1 WHEN ``,
            predictor,
            ``=''Medium'' THEN 2 WHEN ``,
            predictor,
            ``=''High'' THEN 3
                    ELSE -1
                end
                    ) as `temp_predictor`
                from Challenger_schema.',
```

```
            tbl,
            ') converter_tbl
                    ORDER BY `ED` ASC LIMIT ',
            k,
            ';'
        );
        prepare getsql1 from @sql1;
        execute getsql1;
    ELSE
        SELECT 'Error: Value of @predictor is not valid.';
    END IF;

    -- print @k nearest neighbors
    SELECT `O-Ring failure`, `Launch temperature`, `Leak-
check pressure`
    FROM Challenger_schema.temp1;

    -- Return the proportion of N and Y.
    SET @prosql = CONCAT(
        'SELECT `',
        tar,
        '`, CONCAT(ltrim(
            CAST(count(`',
        tar,
        '`)*100.0/(SELECT count(`',
        tar,
        '`) FROM Challenger_schema.temp1) AS DEC(18,2))
        ),
        ''%'') as `proportion`
        FROM Challenger_schema.temp1
        GROUP BY `',
        tar,
        '`;'
    );
    prepare getprosql from @prosql;
    execute getprosql;

    -- Return the majority class (N or Y) of @target
    SET @sub_countsql = CONCAT(
        '(
            SELECT
                `',
        tar,
        '`,
```

```
        count(``,
        tar,
        ``) as `proportion`
        FROM Challenger_schema.temp1
        GROUP BY ``,
        tar,
        ``
        ) count_pro'
    );
    SET @majsql = CONCAT(
        'SELECT ``,
        tar,
        `` FROM ',
        @sub_countsql,
        ' WHERE `proportion` IN ( '
        'SELECT MAX(`proportion`) AS `proportion`
        FROM ',
        @sub_countsql,
        ');'
    );
    prepare getmajsql from @majsql;
    execute getmajsql;

    -- drop temp table
    DROP TABLE IF EXISTS Challenger_schema.temp1;

END
```



3- Find @k nearest neighbors to @value by measuring its distance to values in @predictor column.

Example:

```
CALL `Challenger_schema`.`get_knn`('Challenger', 'O-
Ring failure', 'Launch temperature', '30', 5);
```

O-Ring failure	Launch temperature	Leak-check pressure
1	53	High
1	57	High
1	58	High
1	63	High
0	66	Low

```
CALL `Challenger_schema`.`get_knn`('Challenger', 'O-
Ring failure', 'Leak-check pressure', 'Medium', 3);
```

O-Ring failure	Launch temperature	Leak-check pressure
0	70	Medium
0	73	Medium
1	58	High

4- Return the majority class (N or Y) of @target or the proportion of N and Y.

Example:

```
CALL `Challenger_schema`.`get_knn`('Challenger', 'O-
Ring failure', 'Launch temperature', '30', 5);
```

	O-Ring failure	proportion
▶	1	80.00%
	0	20.00%

	O-Ring failure
▶	1

```
CALL `Challenger_schema`.`get_knn`('Challenger', 'O-
Ring failure', 'Leak-check pressure', 'Medium', 3);
```

	O-Ring failure	proportion
▶	0	66.67%
	1	33.33%

	O-Ring failure
▶	0