School Ranking Analysis

1. Write a query to create a **students** table with the student ID, first name, last name, class, and age fields and ensure that the last name, first name, and student ID fields have the NOT NULL constraint and that the student ID field is a primary key

   SQL code:

```
CREATE TABLE lep_5.students (
    s_id INT NOT NULL,
    s_fname varchar(45) NOT NULL,
    s_lname varchar(45) NOT NULL,
    class varchar(45) NULL,
    age INT NOT NULL,
    PRIMARY KEY(s_id));
```

2. Write a query to create a **marksheet** table with score, year, ranking, class, and student ID fields

   SQL code:

```
CREATE TABLE lep_5.marksheet (
    score INT NOT NULL,
    year INT NULL,
    class varchar(45) NULL,
    ranking varchar(45) NULL,
    s_id INT NOT NULL);
```

3. Write a query to insert values into the students and marksheet tables

   SQL code: Students table

```
INSERT INTO lep_5. students (s_id,s_fname,s_lname,class,age) VALUES
('01','krishna','gee','10','18');
```

SQL code: Marksheet table

INSERT INTO lep_5. marksheet (score,year,class,ranking,s_id) VALUES ('989','2014','10','1','1');

4. Write a query to display the student ID and first name of every student in the **students** table whose age is greater than or equal to 16 and whose last name is Kumar

SQL code:

SELECT s_id,s_fname FROM lep_5.students WHERE s_lname = 'kumar' AND age>=16;

Output:

s_id	s_fname
3	Kailash
11	rosy
NULL	NULL

5. Write a query to display the details of every student from the **marksheet** table whose score is between 800 and 1000

SQL code:

SELECT * FROM lep_5.marksheet WHERE score BETWEEN 800 AND 1000 ;

Output:

score	year	class	ranking	s_id
989	2014	10	1	1
880	2014	10	4	3
870	2014	10	5	4
900	2014	10	3	7
801	2014	10	6	9
970	2014	10	2	11

6. Write a query to increase the score in the **marksheet** table by five and create a new score column to display this new score

SQL code:

```
SELECT * ,score+5 as new_score FROM lep_5.marksheet ;
```

Output:

score	year	class	ranking	s_id	new_score
989	2014	10	1	1	994
454	2014	10	10	2	459
880	2014	10	4	3	885
870	2014	10	5	4	875
720	2014	10	7	5	725
670	2014	10	8	6	675
900	2014	10	3	7	905
540	2014	10	9	8	545
801	2014	10	6	9	806

7. Write a query to display the **marksheet** table in descending order of the score

SQL code:

```
SELECT * FROM lep_5.marksheet ORDER BY score DESC ;
```

Output:

score	year	class	ranking	s_id
989	2014	10	1	1
970	2014	10	2	11
900	2014	10	3	7
880	2014	10	4	3
870	2014	10	5	4
801	2014	10	6	9
720	2014	10	7	5
720	2014	10	12	12
670	2014	10	8	6
540	2014	10	9	8
454	2014	10	10	2
420	2014	10	11	10

8. Write a query to display the **marksheet** table in descending order of the score

SQL code:

```
SELECT * FROM lep_5.students WHERE s_fname LIKE 's%';
```

Output:

s_id	s_fname	s_lname	class	age
2	Stephen	Christ	10	17
7	saurab	kothari	10	15
10	sara	rayan	10	16
NULL	NULL	NULL	NULL	NULL