SMARTBRIDGE EXTERNSHIP

Modern Application Development

Java Spring Boot

Assignment 2

Database Management System Queries

NAME: ARAVINDKRISHNA R

REG. No.: 20BCE0074

DATE: 28-05-2023

1. create, update, delete commands in mysql

Create

Creating table and inserting values,

```
CREATE TABLE Student (
  id INT PRIMARY KEY NOT NULL,
  name VARCHAR(50) NOT NULL,
  age INT
);
```

Output

Action Output

#	Time	Action	Message
1	15:32:45	CREATE TABLE Student (id INT PRIMARY KEY NOT NULL, name VARCHAR(50) NOT NULL, age INT)	0 row(s) affected

```
INSERT INTO Student (id, name, age)
VALUES
(1, 'Aravind', 20),
(2, 'Krishna', 22),
(3, 'Raj', 19);
select * from javaspring.Student
```

Result Grid

id	name	age
1	Aravind	20
2	Krishna	22
3	Raj	19
NULL	NULL	NULL

Output

Action Output

#	Time	Action	Message
1	15:32:45	CREATE TABLE Student (id INT PRIMARY KEY NOT NULL, name VARCHAR(50) NOT NULL, age INT)	0 row(s) affected
2	15:41:56	INSERT INTO Student (id, name, age) VALUES (1, 'Aravind', 20), (2, 'Krishna', 22), (3, 'Raj', 19)	3 row(s) affected Reco
3	15:42:53	select * from javaspring.Student LIMIT 0, 1000	3 row(s) returned

Update

Updating the values

```
UPDATE javaspring.Student
SET age = 21
WHERE id = 2;

Select * from javaspring.Student
```

Automatic context help is disabl
help for the current caret pos

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	name	age
1	Aravind	20
2	Krishna	21
3	Raj	19
NULL	NULL	NULL

Result Grid | Form Editor | Field Types

Student 2 × | Apply | Revert | Context Help | Snippets

Output

Action Output

#	Time	Action	Message
1	15:32:45	CREATE TABLE Student (id INT PRIMARY KEY NOT NULL, name VARCHAR(50) NOT NULL, age INT)	0 row(s) affected
2	15:41:56	INSERT INTO Student (id, name, age) VALUES (1, 'Aravind', 20), (2, 'Krishna', 22), (3, 'Raj', 19)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
3	15:42:53	select * from javaspring.Student LIMIT 0, 1000	3 row(s) returned
4	15:46:55	UPDATE javaspring.Student SET age = 21 WHERE id = 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
5	15:48:51	Select * from javaspring.Student LIMIT 0, 1000	3 row(s) returned

Delete

Deleting the values and the table

```
DELETE FROM Student
WHERE id = 3;

Select * from javaspring.Student
```

Automatic context help is
help for the current care

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	name	age
1	Aravind	20
2	Krishna	21
NULL	NULL	NULL

Result Grid | Form Editor | Field Types

Student 3 × | Apply | Revert | Context Help | Snippets

Output

Action Output

#	Time	Action	Message
2	15:41:56	INSERT INTO Student (id, name, age) VALUES (1, 'Aravind', 20), (2, 'Krishna', 22), (3, 'Raj', 19)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
3	15:42:53	select * from javaspring.Student LIMIT 0, 1000	3 row(s) returned
4	15:46:55	UPDATE javaspring.Student SET age = 21 WHERE id = 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
5	15:48:51	Select * from javaspring.Student LIMIT 0, 1000	3 row(s) returned
6	15:52:20	DELETE FROM Student WHERE id = 3	1 row(s) affected
7	15:52:24	Select * from javaspring.Student LIMIT 0, 1000	2 row(s) returned

2. create tables and perform joins in mysql

2 tables created

```sql
CREATE TABLE Students (
  id INT PRIMARY KEY,
  name VARCHAR(50),
  email VARCHAR(50)
);

CREATE TABLE Courses (
  id INT PRIMARY KEY,
  student_id INT,
  course_name VARCHAR(50)
);
```

Inserting values

```sql
INSERT INTO Students (id, name, email)
VALUES (1, 'Aravind', 'ak@gmail.com');

INSERT INTO Students (id, name, email)
VALUES (2, 'Krishna', 'krish@gmail.com');

INSERT INTO Courses (id, student_id, course_name)
VALUES (1, 1, 'Mathematics');

INSERT INTO Courses (id, student_id, course_name)
VALUES (2, 1, 'Science');

INSERT INTO Courses (id, student_id, course_name)
VALUES (3, 2, 'Tamil');
```

Output

Action Output

#	Time	Action	Message
10	16:20:59	CREATE TABLE Courses (id INT PRIMARY KEY, student_id INT, course_name VARCHAR(50))	0 row(s) affected
11	16:24:09	INSERT INTO Students (id, name, email) VALUES (1, 'Aravind', 'ak@gmail.com')	1 row(s) affected
12	16:24:16	INSERT INTO Students (id, name, email) VALUES (2, 'Krishna', 'krish@gmail.com')	1 row(s) affected
13	16:24:19	INSERT INTO Courses (id, student_id, course_name) VALUES (1, 1, 'Mathematics')	1 row(s) affected
14	16:24:22	INSERT INTO Courses (id, student_id, course_name) VALUES (2, 1, 'Science')	1 row(s) affected
15	16:24:25	INSERT INTO Courses (id, student_id, course_name) VALUES (3, 2, 'Tamil')	1 row(s) affected

```sql
INSERT INTO Students (id, name, email)
VALUES (3, 'Raj', 'Raj@gmail.com');
```

Result

Students

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	name	email
1	Aravind	ak@gmail.com
2	Krishna	krish@gmail.com
3	Raj	Raj@gmail.com
NULL	NULL	NULL

Courses

```sql
Select * from javaspring.Courses;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	student_id	course_name
1	1	Mathematics
2	1	Science
3	2	Tamil
NULL	NULL	NULL

Joins

Inner Join

SQL File 4* | Limit to 1000 rows

```sql
SELECT *
FROM Students
INNER JOIN Courses ON  Students.id=Courses.student_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	name	email	id	student_id	course_name
1	Aravind	ak@gmail.com	1	1	Mathematics
1	Aravind	ak@gmail.com	2	1	Science
2	Krishna	krish@gmail.com	3	2	Tamil

Left Join

```
SELECT *
FROM Students
Left JOIN Courses ON  Students.id=Courses.student_id;
```

Result Grid

id	name	email	id	student_id	course_name
1	Aravind	ak@gmail.com	2	1	Science
1	Aravind	ak@gmail.com	1	1	Mathematics
2	Krishna	krish@gmail.com	3	2	Tamil
3	Raj	Raj@gmail.com	NULL	NULL	NULL

Right Join

SQL File 4*

```
SELECT *
FROM Students
Right JOIN Courses ON  Students.id=Courses.student_id;
```

Result Grid

id	name	email	id	student_id	course_name
1	Aravind	ak@gmail.com	1	1	Mathematics
1	Aravind	ak@gmail.com	2	1	Science
2	Krishna	krish@gmail.com	3	2	Tamil

Full Join

as give in git material

SQL File 4*

```
SELECT *
FROM Students
Full JOIN Courses
```

Result Grid

id	name	email	id	student_id	course_name
3	Raj	Raj@gmail.com	1	1	Mathematics
2	Krishna	krish@gmail.com	1	1	Mathematics
1	Aravind	ak@gmail.com	1	1	Mathematics
3	Raj	Raj@gmail.com	2	1	Science
2	Krishna	krish@gmail.com	2	1	Science
1	Aravind	ak@gmail.com	2	1	Science
3	Raj	Raj@gmail.com	3	2	Tamil
2	Krishna	krish@gmail.com	3	2	Tamil
1	Aravind	ak@gmail.com	3	2	Tamil

another one (normal union method)

SQL File 4*

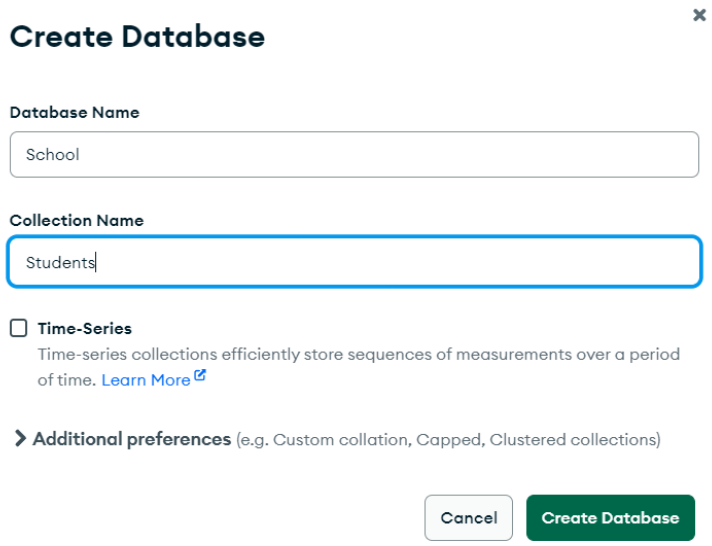
```
SELECT Students.id, Students.name, Students.email, Courses.course_name
FROM Students
LEFT JOIN Courses ON Students.id = Courses.student_id
UNION
SELECT Courses.id, Students.name, Students.email, Courses.course_name
FROM Courses
RIGHT JOIN Students ON Courses.student_id = Students.id;
```

Result Grid

id	name	email	course_name
1	Aravind	ak@gmail.com	Science
1	Aravind	ak@gmail.com	Mathematics
2	Krishna	krish@gmail.com	Tamil
3	Raj	Raj@gmail.com	NULL
2	Aravind	ak@gmail.com	Science
3	Krishna	krish@gmail.com	Tamil
NULL	Raj	Raj@gmail.com	NULL

3. create, update, delete commands in mongo

Create



School database and Student collection is created,

```
> show dbs
< School   8.00 KiB
  admin   40.00 KiB
  config  72.00 KiB
  local   72.00 KiB
  sample   8.00 KiB
test>
```

selecting that School database,

```
> use School
< switched to db School
School>
```

Creating one more collection,

```
> db.createCollection("Teachers")
< { ok: 1 }
School>
```

Inserting data into Students, we can also insert in teachers (Creating documents) using insertOne,

```
> db.Students.insertOne({name: "Aravind", age: 20})
< {
    acknowledged: true,
    insertedId: ObjectId("6473098eaf43f055a81c9f67")
  }
> db.Students.find()
< {
    _id: ObjectId("6473098eaf43f055a81c9f67"),
    name: 'Aravind',
    age: 20
  }
School >
```

we can also use insertMany

```
> db.Students.insertMany([{name: "Krishna", age: 21},{name: "Raj", age: 25}])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId("64730bacaf43f055a81c9f68"),
      '1': ObjectId("64730bacaf43f055a81c9f69")
    }
  }
> db.Students.find()
< {
    _id: ObjectId("6473098eaf43f055a81c9f67"),
    name: 'Aravind',
    age: 20
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f68"),
    name: 'Krishna',
    age: 21
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f69"),
    name: 'Raj',
    age: 25
  }
School >
```

Update - Updating the values using updateOne,

```
> db.Students.updateOne({name:"Aravind"}, {$set: {age: 19}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.Students.find()
< {
    _id: ObjectId("6473098eaf43f055a81c9f67"),
    name: 'Aravind',
    age: 19
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f68"),
    name: 'Krishna',
    age: 21
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f69"),
    name: 'Raj',
    age: 25
  }
School >
```

Updating the values using updateMany,

```
> db.Students.updateMany({age: {$gt:20}}, {$set: {age: 26}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 2,
    modifiedCount: 2,
    upsertedCount: 0
  }
> db.Students.find()
< {
    _id: ObjectId("6473098eaf43f055a81c9f67"),
    name: 'Aravind',
    age: 19
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f68"),
    name: 'Krishna',
    age: 26
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f69"),
    name: 'Raj',
    age: 26
  }
School >
```

Delete

deleting the values using deleteOne,

```
> db.Students.deleteOne({name: "Krishna"})
< {
    acknowledged: true,
    deletedCount: 1
  }
> db.Students.find()
< {
    _id: ObjectId("6473098eaf43f055a81c9f67"),
    name: 'Aravind',
    age: 19
  }
  {
    _id: ObjectId("64730bacaf43f055a81c9f69"),
    name: 'Raj',
    age: 26
  }
School >
```

deleting the values using deleteMany,

```
> db.Students.deleteMany({age: {$lt: 30}})
< {
    acknowledged: true,
    deletedCount: 2
  }
> db.Students.find()
<
School >
```

deleting the collection and the database,

```
> db.Students.drop()
< true
```

```
> db.Students.drop()
< true
> db.dropDatabase()
< { ok: 1, dropped: 'School' }
```