← Project review - SQL beginner. Day08

Type of project	Individual
Duration	30 min
Passed Peer Reviews	1/2

Git project

ssh://git@repos-ssh.21-school.ru:2289/students/SQL_beginner._Day08.ID_574106/mma...

Copy link Open

Student

mmarcele-sql-tester level 2

About

Introduction

The methodology of School 21 makes sense only if peer-to-peer reviews are done seriou sly. Please read all guidelines carefully before starting the review.
- Please, stay courteous, polite, respectful and constructive in all communications duri ng this review.
- Highlight possible malfunctions of the work done by the person and take the time to discuss and debate it.

- Keep in mind that sometimes there can be differences in interpretation of the tasks an d the scope of features. Please, stay open-minded to the vision of the other.
- If you have not finished the project yet, it is compulsory to read the entire instruction before starting the review.

Guidelines

- Evaluate only the files that are in src folder on the GIT repository of the student or gro up.
- Ensure to start reviewing a group project only when the team is present in full.
- Use special flags in the checklist to report, for example, an “empty work” if repository does not contain the work of the student (or group) in the src folder of the develop bra nch, or “cheat” in case of cheating or if the student (or group) are unable to explain the ir work at any time during review as well as if one of the points below is not met. Howe ver, except for cheating cases, you are encouraged to continue reviewing the project to identify the problems that caused the situation in order to avoid them at the next revie w.
- Doublecheck that the GIT repository is the one corresponding to the student or the gr oup.
- Meticulously check that nothing malicious has been used to mislead you.
- In controversial cases, remember that the checklist determines only the general order of the check. The final decision on project evaluation remains with the reviewer.

Main part

Exercise 00

Checks for the file day08_ex00.sql
- The SQL scripts look like below.

```
-- Session #1
BEGIN;

update pizzeria
```

```
set rating = 5
where name = 'Pizza Hut';

select *
from pizzeria
where name  = 'Pizza Hut';


COMMIT;

--Session #2
select *
from pizzeria
where name  = 'Pizza Hut';
```

- The result of Session #1

```
=> begin;
BEGIN
=*> update pizzeria set rating = 5 where name = 'Pizza Hut';
UPDATE 1
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    5
(1 row)
=*> commit;
COMMIT
```

- The result of Session #2

```
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    4.6
(1 row)
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    5
(1 row)
```

No Yes

Exercise 01

Checks for the file day08_ex01.sql
- The SQL script looks like below.

```
-- Session #1
begin transaction isolation level read committed;
select * from pizzeria where  name  = 'Pizza Hut';
update pizzeria set rating = 4 where name  = 'Pizza Hut';

commit;
select * from pizzeria where  name  = 'Pizza Hut';

--Session #2
begin transaction isolation level read committed;
select * from pizzeria where  name  = 'Pizza Hut';
update pizzeria set rating = 3.6 where name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';
```

- The result of Session #1

```
=> begin transaction isolation level read committed;
BEGIN
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    5
(1 row)
=*> update pizzeria set rating = 4 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
```

- The result of Session #2

```
=> begin transaction isolation level read committed;
BEGIN
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    5
(1 row)
=*> update pizzeria set rating = 3.6 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
```

```
COMMIT
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
```

No Yes

Exercise 02

Checks for the file day08_ex02.sql
- The SQL script looks like below.

```
-- Session #1
begin transaction isolation level repeatable read;
select * from pizzeria where  name  = 'Pizza Hut';
update pizzeria set rating = 4 where name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';

--Session #2
begin transaction isolation level repeatable read;
select * from pizzeria where  name  = 'Pizza Hut';
update pizzeria set rating = 3.6 where name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';
```

- The result of Session #1

```
=> begin transaction isolation level repeatable read ;
BEGIN
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
=*> update pizzeria set rating = 4 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    4
(1 row)
```

- The result of Session #2

```
=> begin transaction isolation level repeatable read ;
BEGIN
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating

---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
=*> update pizzeria set rating = 3.6 where name = 'Pizza Hut';
ERROR: could not serialize access due to concurent update
=!> commit;
ROLLBACK
=*> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    4
(1 row)
```

No Yes

Exercise 03

Checks for the file day08_ex03.sql
- The SQL script looks like below.

```
-- Session #1
begin transaction isolation level read committed;
select * from pizzeria where  name  = 'Pizza Hut';
select * from pizzeria where  name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';

--Session #2
begin transaction isolation level read committed;
update pizzeria set rating = 3.6 where name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';
```

- The result of Session #1

```
=> begin transaction isolation level read committed ;
BEGIN
=*> select * from pizzeria where name = 'Pizza Hut';
```

```
    id |   name    | rating
    ---+-----------+--------
    1  |Pizza Hut  |    4
    (1 row)
    =*> select * from pizzeria where name = 'Pizza Hut';
    id |   name    | rating
    ---+-----------+--------

    1  |Pizza Hut  |    3.6
    (1 row)
    =*> commit;
    COMMIT
    => select * from pizzeria where name = 'Pizza Hut';
    id |   name    | rating
    ---+-----------+--------
    1  |Pizza Hut  |    3.6
    (1 row)
```

- The result of Session #2

```
    => begin transaction isolation level read committed ;
    BEGIN
    =*> update pizzeria set rating = 3.6 where name = 'Pizza Hut';
    UPDATE 1
    =*> commit;
    COMMIT
    => select * from pizzeria where name = 'Pizza Hut';
    id |   name    | rating
    ---+-----------+--------
    1  |Pizza Hut  |    3.6
    (1 row)
```

No Yes

Exercise 04

Checks for the file day08_ex04.sql
- The SQL script looks like below.

```
    -- Session #1
    begin transaction isolation level serializable;
    select * from pizzeria where  name  = 'Pizza Hut';
    select * from pizzeria where  name  = 'Pizza Hut';
    commit;
    select * from pizzeria where  name  = 'Pizza Hut';
```

```
--Session #2
begin transaction isolation level serializable;
update pizzeria set rating = 3.6 where name  = 'Pizza Hut';
commit;
select * from pizzeria where  name  = 'Pizza Hut';
```

- The result of Session #1

```
=> begin transaction isolation level serializable ;
BEGIN
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.6
(1 row)
=> commit;
COMMIT
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.0
(1 row)
```

- The result of Session #2

```
=> begin transaction isolation level serializable ;
BEGIN
=*> update pizzeria set rating = 3.6 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
=*> begin transaction isolation level serializable ;
BEGIN
=*> update pizzeria set rating = 3.0 where name = 'Pizza Hut';
UPDATE 1
=> commit;
COMMIT
=> select * from pizzeria where name = 'Pizza Hut';
id |   name    | rating
---+-----------+--------
1  |Pizza Hut  |    3.0
(1 row)
```

No Yes

Exercise 05

Checks for the file day08_ex05.sql
- The SQL script looks like below.

```
 -- Session #1
 begin transaction isolation level read committed;
 select sum(rating) from pizzeria;
 select sum(rating) from pizzeria;
 commit;
 select sum(rating) from pizzeria;

 --Session #2
 begin transaction isolation level read committed;
 update pizzeria set rating = 1 where name  = 'Pizza Hut';
 commit;
 select sum(rating) from pizzeria;
```

- The result of Session #1

```
=> begin transaction isolation level read committed ;
BEGIN
=*> select sum(rating)from pizzeria;
sum
---
21.9
(1 row)
=*> select sum(rating)from pizzeria;
sum
---
19.9
(1 row)
=> commit;
COMMIT
=> select sum(rating)from pizzeria;
sum
---
19.9
(1 row)
```

- The result of Session #2

```
=> begin transaction isolation level read committed ;
```

```
BEGIN
=*> update pizzeria set rating = 1 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
=> select sum(rating) from pizzeria;
sum
---
19.9
(1 row)
```

No Yes

Exercise 06

Checks for the file day08_ex06.sql
- The SQL script looks like below.

```
-- Session #1
begin transaction isolation level repeatable read;
select sum(rating) from pizzeria;
select sum(rating) from pizzeria;
commit;
select sum(rating) from pizzeria;

--Session #2
begin transaction isolation level repeatable read;
update pizzeria set rating = 5 where name  = 'Pizza Hut';
commit;
select sum(rating) from pizzeria;
```

- The result of Session #1

```
=> begin transaction isolation level repeatable read ;
BEGIN
=*> select sum(rating)from pizzeria;
sum
---
19.9
(1 row)
=*> select sum(rating)from pizzeria;
sum
---
19.9
(1 row)
=*> commit;
COMMIT
```

```
=*> select sum(rating)from pizzeria;
sum
---
23.9
(1 row)
```

- The result of Session #2

```
=> begin transaction isolation level repeatable read ;
BEGIN
=*> update pizzeria set rating = 5 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
=> select sum(rating)from pizzeria;
sum
```

No Yes

Exercise 07

Checks for the file day08_ex07.sql
- The SQL script looks like below.

```
-- Session #1
begin;
update pizzeria set rating = 4 where name = 'Dominos';
update pizzeria set rating = 4 where name = 'Pizza Hut';
commit;
select sum(rating) from pizzeria;

--Session #2
begin;
update pizzeria set rating = 3 where name = 'Pizza Hut';
update pizzeria set rating = 3 where name = 'Dominos';
commit;
```

- The result of Session #1

```
=> begin;
BEGIN
=*> update pizzeria set rating = 4 where name = 'Dominos';
```

```
UPDATE 1
=*> update pizzeria set rating = 4 where name = 'Pizza Hut';
UPDATE 1
=*> commit;
COMMIT
```

- The result of Session #2 (should be error is deadlock detected)

```
=> begin;
BEGIN
=*> update pizzeria set rating = 4 where name = 'Pizza Hut';
UPDATE 1
=*> update pizzeria set rating = 4 where name = 'Dominos';
ERROR: deadlock detected
DETAIL: ...
=*> commit;
ROLLBACK
```

No Yes

Feedback

Fails

Empty work Forbidden functions Cheat

Comment

Leave a comment...

Review