MONASH University

FIT2095 e-Business software technologies - S2 2021

Dashboard / My units / FIT2095_S2_2021 / Assessments / Week 6 Pre-Reading Quiz

Started on	Sunday, 29 August 2021, 9:42 PM
State	Finished
Completed on	Sunday, 29 August 2021, 9:49 PM
Time taken	6 mins 46 secs
Grade	**6.00** out of 10.00 (**60**%)

Print friendly format

Question **1**

Correct

Mark 1.00 out of 1.00

Retrieve the first 50 documents with quantity between 100 and 150 inclusive.

Which of the following statements fulfills the above and is correct?

Select one:

○ a.
```
Items.where('quantity').gte(100).lte(150).sort(50).exec(function (err, docs) {
 // Do something with Docs
});
```

○ b.
```
Items.where('quantity').gte(100).lte(150).limit(50).function (err, docs) {
 // Do something with Docs
};
```

○ c.
```
Items.where('quantity').gt(100).lt(150).limit(50).exec(function (err, docs) {
 // Do something with Docs
});
```

● d.
```
Items.where('quantity').gte(100).lte(150).limit(50).exec(function (err, docs) {
 // Do something with Docs
});
```
✓

Your answer is correct.

The correct answer is:
```
Items.where('quantity').gte(100).lte(150).limit(50).exec(function (err, docs) {
 // Do something with Docs
});
```

Question **2**

Incorrect

Mark 0.00 out of 1.00

Develop a schema for items with two properties: itemName (String) and quantity (Integer). The quantity should be saved if it is Integer (not decimal).

Which of the following fulfills the above requirements and is correct?

Hint: https://www.w3schools.com/jsref/jsref_isinteger.asp

Select one:

● a. Declaring the quantity field data type as Number would be enough. ✖

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: Number
});
```

○ b.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
        type: Number,
        validate: {
              validator: Number.isInteger,
              message: 'The quantity is not an integer value'
             }
     }
});
```

○ c.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
        type: Number,
        validate: {
              validator: function(quantity){Number.isInteger(quantity);},
              message: 'The quantity is not an integer value'
             }
     }
});
```

○ d.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
        type: Number,
        validate: {
              validator: isInteger,
              message: 'The quantity is not an integer value'
             }
     }
});
```

Your answer is incorrect.

The correct answer is:

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
        type: Number,
        validate: {
              validator: Number.isInteger,
              message: 'The quantity is not an integer value'
             }
     }
});
```

Question **3**

Correct

Mark 1.00 out of 1.00

```
let author1 = new Author({
 _id: new mongoose.Types.ObjectId(),
 name: {
      firstName: 'Tim',
      lastName: 'John'
   },
 age: 80
});
```

Find all the documents with the first name equals 'Tim'. Which of the following is correct?

Select one:

○ a.
```
Items.findOne({ 'firstName': 'Tim' }, function (err, docs) {
   //Do something
});
```

○ b.
```
Items.find({ 'name.firstName': /^Tim/ }, function (err, docs) {
   //Do something
});
```

○ c.
```
Items.find({ name.firstName: 'Tim' }, function (err, docs) {
   //Do something
});
```

● d. ✓
```
Items.find({ 'name.firstName': 'Tim' }, function (err, docs) {
   //Do something
});
```

○ e.
```
Items.find({ 'name.firstname': 'Tim' }, function (err, docs) {
   //Do something
});
```

Your answer is correct.

The correct answer is:
```
Items.find({ 'name.firstName': 'Tim' }, function (err, docs) {
   //Do something
});
```

Question **4**

Incorrect

Mark 0.00 out of 1.00

The following statement creates a new Model for the 'itemSchema' schema.

```
module.exports = mongoose.model('Items', itemSchema);
```

What does 'Items' represent?

Select one:

● a. The name of the database the schema will connect to ✗

○ b. The name of the server that hosts the database

○ c. The name of the items' collection

○ c. The name of the items' collection

○ d. The name of the primary key column

○ e. Just a unique name to identify the schema

Your answer is incorrect.

The correct answer is: The name of the items' collection

Question 5

Correct

Mark 1.00 out of 1.00

Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js.

With respect to Mongoose, which of the following statements is false?

Select one:

● a. In order to use Mongoose, you must have a reference to MongoDB module. ✓

```
const mongodb=require('MongoDB');
const mongoose = require('mongoose');
```

○ b. Query functions -such as findOne and UpdateMany- are accessible through Models, not schemas.

○ c. Using Mongoose, you can build your own validation function.

○ d. Using Mongoose, you can strongly-typed MongoDB collections.

○ e. With Mongoose, I am able to implement One-To-Many relationship between collections

Your answer is correct.

The correct answer is: In order to use Mongoose, you must have a reference to MongoDB module.

```
const mongodb=require('MongoDB');
const mongoose = require('mongoose');
```

Question 6

Correct

Mark 1.00 out of 1.00

Develop a schema for two properties: itemName (String) and quantity (Number) where the item quantity is a positive number.

Which of the following statements fulfills the above requirements and is correct?

Select one:

○ a.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: {
                    validator: function (newQuantity) { return newQuantity < 0;},
                    message: 'The value is not positive'
                  }
           }
  });
```

● b. ✓

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: {
                    validator: function (newQuantity) { return newQuantity >= 0;},
                    message: 'The value is not positive'
                  }
           }
  });
```

○ c.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: {
                    function (newQuantity) { return newQuantity >= 0;},
                    message: 'The value is not positive'
                  }
           }
  });
```

○ d.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: {
                    validator: function (newQuantity) { if(newQuantity >= 0) console.log('It is not positive);}
                  }
           }
  });
```

○ e.

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: function (newQuantity) { return newQuantity >= 0};
      }
});
```

Your answer is correct.

The correct answer is:

```
var itemSchema = mongoose.Schema({
    _id: mongoose.Schema.Types.ObjectId,
    itemName: String,
    quantity: {
          type: Number,
          validate: {
                    validator: function (newQuantity) { return newQuantity >= 0;},
                    message: 'The value is not positive'
                  }
           }
  });
```

Question **7**

Correct

Mark 1.00 out of 1.00

Develop a Mongoose schema that consists of two properties: item name (string) and quantity (integer). The quantity should be required and has a default value = 0.

which of the following fulfills the above requirements and correct?

Select one:

○ a.

```
var itemSchema = mongoose.Schema({
```

○ a.

```
var itemSchema = mongoose.Schema({
   _id: mongoose.Schema.Types.ObjectId,
   itemName: String,
   quantity: {
          type: Number,
          required: 'True',
          default: 0
        }
  });
```

● b. ✓

```
var itemSchema = mongoose.Schema({
   _id: mongoose.Schema.Types.ObjectId,
   itemName: String,
   quantity: {
          type: Number,
          required: true,
          default: 0
        }
  });
```

○ c.

```
var itemSchema = MongoDB.Schema({
   _id: mongoose.Schema.Types.ObjectId,
   itemName: String,
   quantity: {
          type: Number,
          required: true,
          default: 0
        }
  });
```

○ d.

```
var itemSchema = mongoose.Schema({
   _id: mongoose.Schema.Types.Id,
   itemName: String,
   quantity: {
          type: Integer,
          required: true,
          default: 0
        }
  });
```

○ e.

```
var itemSchema = mongoose.Schema({
   _id: mongoose.Schema.Types.ObjectId,
   itemName: String,
   quantity: {
          type: Integer,
          required: true,
          default: 0
        }
  });
```

Your answer is correct.

The correct answer is:

```
var itemSchema = mongoose.Schema({
   _id: mongoose.Schema.Types.ObjectId,
   itemName: String,
   quantity: {
          type: Number,
          required: true,
          default: 0
        }
  });
```

Question **8**

Correct

Mark 1.00 out of 1.00

```
let author1 = new Author({
       _id: new mongoose.Types.ObjectId(),
       name: {
            firstName: 'Tim',
            lastName: 'John'
       },
       age: 80
 });

 author1.save(function (err) {
        if (err) throw err;

        console.log('Author successfully Added to DB');
  };

var book1 = new Book({
      _id: new mongoose.Types.ObjectId(),
      title: 'FIT2095 Book ',
      author: author1._id,
       isbn: '123456',
   });

book1.save(function (err) {
 if (err) throw err;

 console.log('Book1 successfully Added to DB');
 });
```

With respect to the presented code, which of the following is true?

Select one:

● a. The code has an issue. ✓

The creating and saving of book1 should be done inside author1.save() callback function due to the asynchronous call of I/O operations

○ b. The code is perfect and has no issue

○ c. The code has an issue.

Both author1 and book1 will be saved in the same collection.

○ d. The code a syntax error.

It uses 'var' instead of 'let'

Your answer is correct.

The correct answer is: The code has an issue.
The creating and saving of book1 should be done inside author1.save() callback function due to the asynchronous call of I/O operations

Question 9

Incorrect

Mark 0.00 out of 1.00

Assume you have this Mongoose URL:

```
let url='mongodb://localhost:27017/Travel';
```

What does the word 'Travel' represent?

Select one:

● a. Collection name ✖

○ b. last part of the server's address

○ c. Database name

○ d. MongoDB replica name

Your answer is incorrect.

The correct answer is: Database name

Question **10**

Incorrect

Mark 0.00 out of 1.00

Which of the following is false about the presented code?

```
let bookSchema = mongoose.Schema({
 _id: mongoose.Schema.Types.ObjectId,
     title: {
             type: String,
             required: true
           },
     isbn: String,
     author: {
             type: mongoose.Schema.Types.ObjectId,
             ref: 'Author'
           },
     created: {
            type: Date,
            default: Date.now
          }
});
```

Select one:

● a. the field isbn is optional ✖

○ b. the presented code declares a schema without a Model

○ c. the field author is a reference to another schema (document)

○ d. the field **created** is mandatory

Your answer is incorrect.

The correct answer is: the field **created** is mandatory

◄ Week 5 Pre-Reading Quiz

Jump to...

Week 7 Pre-Reading Quiz ►