# Databases Advanced Exam: Photography Workshops

# Data Model Definition

The Japanese photographer Fo To has decided to start organizing and managing workshops for photography enthusiasts. He has a big set of gear(lenses, cameras, accessories) that he provide for given workshop. He need to keep information about the workshops, photographers and the equipment. He is not so good at database designing so he asked you to **model a database** for him. Mr Fo want to keep information about:

##### Lens

* **Make**
* **Focal Length –** integer number, represents the focal lentgth in milimeters
* **Max Aperture –** floating point number precise 1 digit after decimal point
* **Compatible With – make of the camera** that the lens is compatible with
* **Owner –** could be **any photographer**

##### DSLR Camera

* **Make –** mandatory information
* **Model –** mandatory information
* **Is Full Frame or Not**
* **Min ISO** – integer number that **cannot be lower than 100 or not set.** Mandatory information
* **Max ISO**
* **Max Shutter Speed –** integer number

##### Mirrorless Camera

* **Make –** mandatory information
* **Model –** mandatory information
* **Is Full Frame or Not**
* **Min ISO** – integer number that **cannot be lower than 100 or not set.** Mandatory information
* **Max ISO**
* **Max Video Resolution –** will be inserted as plain text
* **Max frame rate –** integer number

##### Accessory

* **Name**
* **Owner –** could be any **photographer**

##### Photographer

* **First Name –** mandatory information
* **Last Name** – text that can contain **between 2 and 50 characters,** mandatory information
* **Phone** – must be in the format “+[country\_code]/[phone]” where [country\_code] is **between 1 and 3 digits** and the [phone] is **between 8 and 10 digits.** 
  + Valid phones: +359/88888888, +1/1234579284
  + Invalid phones: -359/88888888, +359/ 4 5 4444444, +359888585313, +412\34553363587
* **Primary Camera –** can be **any camera** (DSLR or Mirrorless). Mandatory information
* **Secondary Camera –** can be **any camera** (DSLR or Mirrorless). Mandatory information
* **Many Lenses**
* **Many Accessories**

##### Workshop

* **Name –** mandatory information
* **Start date**
* **End date**
* **Location –** mandatory information
* **Price Per Participant –** mandatory information
* **Trainer** - any **photographer.** Mandatory information
* **Participants** – many **photographers**

When a **field** is **not told to be mandatory** that means it **can be null**.

Each **photographer** can participate in **many workshops** and in each workshop **many photographers** can participate. Also a workshop has only **one photographer trainer** and **any photogorahper** can be trainer in **many workshops**.

Design a database for Fo using the **Hibernate** and **Code First** approach. You will also need to write several data-driven applications in Java for importing, querying and exporting data from the database.

# Importing Data

Fo provided you some **JSON** and **XML** files to **populate the database with some data**. Import all the data from those files into the database. When **importing lenses and accessories, no data transfer object is required** (but it is recommended). But for all other imports **use data transfer objects.**

You are **not allowed to modify** the provided **JSON** and **XML** files.

### Importing Data from JSON

#### Import lenses

Using the file **lenses.json** create a console application that **imports the data from that file into the database** you created in the previous section. After each iteration of importing lens **print information of the imported lens** in the format provided in the example below.

### Example

#### Input

|  |
| --- |
| **lenses.json** |
| [  { "make" : "Canon", "focalLength" : 50, "maxAperture" : 1.8,"CompatibleWith" : "Canon" },  { "make" : "Nikon", "focalLength" : 35, "maxAperture" : 2.0,"CompatibleWith" : "Nikon" },  { "make" : "Tamron", "focalLength" : 24, "maxAperture" : 2.8,"CompatibleWith" : "Canon" },  { "make" : "Tokina", "focalLength" : 85, "maxAperture" : 1.4,"CompatibleWith" : "Nikon" },  { "make" : "Tokina", "focalLength" : 100, "maxAperture" : 1.2,"CompatibleWith" : "Pentax" },  ] |

#### Output

|  |
| --- |
| Successfully imported Canon 50mm f1.8  Successfully imported Nikon 35mm f2.0  Successfully imported Tamron 24mm f2.8  Successfully imported Tokina 85mm f1.4  Successfully imported Tokina 100mm f1.2 |

#### Import cameras

Using the file cameras.json create a console application that **imports the data from that file into the database**. After each iteration of importing camera print whether the camera was succesfully added to the database.

A **valid recod** for import must contain at least camera’s **type, make, model** and **minimum ISO**. If any of those elements is missing this camera must be ignored and not inserted into the database.

### Example

#### Input

|  |
| --- |
| **cameras.json** |
| [  {  "type" : "DSLR",  "make" : "Canon",  "model" : "500D",  "isFullFrame" : false,  "minISO" : 100,  "maxISO" : 12800,  "MaxShutterSpeed" : 4000  },  {  "type" : "Mirrorless",  "make" : "Sony",  "model" : "A55",  "isFullFrame" : false,  "minISO" : 100,  "maxISO" : 25600,  "maxVideoResolution" : "1920x1080",  "maxFrameRate" : 60  },  {  "type" : "Mirrorless",  "make" : "Sony",  "model" : "A50",  "minISO" : 200  },  {  "type" : "DSLR",  "make" : "Nikon",  "model" : "D3100",  "minISO" : 100  },  {  "type" : "DSLR",  "make" : "Canon",  "minISO" : 40  },  {  "type" : "Mirrorless",  "model" : "A55"  }  ] |

#### Output

|  |
| --- |
| Successfully imported DSLR Canon 500D  Successfully imported Mirrorless Sony A55  Successfully imported Mirrorless Sony A50  Successfully imported DSLR Nikon D3100  Error. Invalid data provided  Error. Invalid data provided |

#### Import photographers

Using the file photographers.json create a console application that imports the data from that file into the database. Select **random primary and secondary camera** from already imported in the database.

In the JSON file the **lenses** for the particular photograph are represented as **IDs of already imported lenses**. If the **ID** of a lens is **not valid or the lens is not compatible with the make** **of the photographer’s primary or secondary camera skip that ID** and continue with trying to add the next one in the array.

After each iteration of importing photographer print whether he/she was succesfully added to the database.

A **valid recod** for import must contain at least **photographer’s first and last name**. If any of those elements is missing this photographer **must be ignored** and not inserted into the database. If phone is present it must be validated otherwise can be null.

### Example

#### Input

|  |
| --- |
| **photographers.json** |
| [  {  "firstName" : "Jared",  "lastName" : "Polin",  "phone" : "+1/12345678",  "lenses" : [1, 2, 4, 5]  },  {  "firstName" : "Samuel",  "lastName" : "Fosso",  "phone" : "+33/12365673",  "lenses" : [8, 6, -5]  },  {  "firstName" : "Julie",  "lastName" : "Moss",  "phone" : "+12459/123673215",  "lenses" : [3, 9]  },  {  "firstName" : "Zhang",  "lastName" : "Xiao",  "phone" : "+44/90645070",  "lenses" : [1231232568]  },  {  "firstName" : "Joseph",  "phone" : "+12459/123673215",  "lenses" : [15, 16]  },  {  "firstName" : "Alberto",  "lastName" : "Korda",  "lenses" : [10]  }  ] |

#### Output

|  |
| --- |
| Successfully imported Jared Polin | Lenses: 4  Successfully imported Samuel Fosso | Lenses: 2  Error. Invalid data provided  Successfully imported Zhang Xiao | Lenses: 0  Error. Invalid data provided  Successfully imported Alberto Korda | Lenses: 1 |

### Importing Data from XML

#### Import accessories

Using the file accessories.xml create a console application that **imports the data from that file into the database** you created in the previous section. After each iteration of importing accessory print information of the imported accessory in the format provided in the example below. Then, for each accessory **assign random photographer owner**, from already imported in the database.

### Example

#### Input

|  |
| --- |
| **accessories.xml** |
| <?xml version="1.0" encoding="utf-8"?>  <accessories>  <accessory name="Tripod Manfrotto 123X" />  <accessory name="Lens Hood E62" />  <accessory name="Lens Hood E77" />  <accessory name="LensPen Pro" />  <accessory name="Lens Cleaning Kit Z220" />  </accessories> |

#### Output

|  |
| --- |
| Successfully imported Tripod Manfrotto 123X  Successfully imported Lens Hood E62  Successfully imported Lens Hood E77  Successfully imported LensPen Pro  Successfully imported Lens Cleaning Kit Z220 |

#### Import workshops

Using the file workshops.xml create a console application that **imports the data from that file into the database** you created in the previous section. After each iteration of importing workshop **print information of the imported workshop in the format provided in the example below**.

The **names of the trainers and participants** would **always be valid names** of photographers already impored in the database. The **name of the trainer** is formed by concatenating his **first name, single space and last name**. If there are any participants enrolled in the workshop they would be always valid.

A **valid recod for import** must contain at least **workshop name**, **location,**  **price** and **trainer**. If any of those elements is missing this workshop **must be ignored** and not inserted into the database.

### Example

#### Input

|  |
| --- |
| **workshops.xml** |
| <?xml version="1.0" encoding="utf-8"?>  <workshops>  <workshop name="Wildlife Birds" start-date="2016-12-04T00:00:00" end-date="2016-12-06T00:00:00" location="Rhodopes" price="120.00">  <trainer>Jared Polin</trainer>  <participants>  <participant first-name="Yan" last-name="Greenidge" />  <participant first-name="Odis" last-name="Borne" />  <participant first-name="Delta" last-name="Tiano" />  </participants>  </workshop>  <workshop start-date="2016-12-09T10:00:00" end-date="2016-12-09T18:00:00" location="Plovdiv" price="89.00">  <trainer>Jared Polin</trainer>  <participants>  <participant first-name="Turla" last-name="Cowger" />  <participant first-name="Kraig" last-name="Velarde" />  </participants>  </workshop>  <workshop name="Polar Lights" start-date="2016-12-05T00:00:00" end-date="2016-12-07T00:00:00" price="930.00">  <trainer>Alberto Korda</trainer>  <participants>  <participant first-name="Bong" last-name="Mungin" />  </participants>  </workshop>  <workshop name="Babies" start-date="2016-12-06T10:30:00" end-date="2016-12-06T14:30:00" location="Sofia" price="40.00">  <participants>  <participant first-name="Delorse" last-name="Bunge" />  </participants>  </workshop>  <workshop name="Street Photography" location="Sofia" price="70.00">  <trainer>Samuel Fosso</trainer>  <participants>  <participant first-name="Amparo" last-name="Goers" />  <participant first-name="Leon" last-name="Recio" />  <participant first-name="Alica" last-name="Lacher" />  <participant first-name="Bunny" last-name="Nolan" />  <participant first-name="Lyle" last-name="Longworth" />  </participants>  </workshop>  <workshop name="Street Photography Part 2" location="Sofia">  <trainer>Samuel Fosso</trainer>  <participants>  <participant first-name="Hungo" last-name="Poter" />  <participant first-name="Camaru" last-name="Kolya" />  </participants>  </workshop>  <workshop name="Street Photography Part 3" location="Sofia" price="90.00">  <trainer>Samuel Fosso</trainer>  </workshop>  </workshops> |

#### Output

|  |
| --- |
| Successfully imported Wildlife Birds  Error. Invalid data provided  Error. Invalid data provided  Error. Invalid data provided  Successfully imported Street Photography  Error. Invalid data provided  Successfully imported Street Photography Part 3 |

# Exporting Data

Fo was so excited that you managed to import his data that he decided to give you some more work to do. He need to **export** some data in **JSON** and **XML** format.

### Exporting to JSON

#### Ordered Photographers

Write a program that creates file photographers-ordered.json that would contain **first names**, **last names** and **phones** of **all photographers** in the database **ordered by first name alphabetically** and by **last name descending**.

### Example

#### Output

|  |
| --- |
| **photographers-ordered.json** |
| [  {  "FirstName": "Adria",  "LastName": "Mcquire",  "Phone": "+123/1234567890"  },  {  "FirstName": "Aimee",  "LastName": "Ochoa",  "Phone": "+123/1234567890"  },  {  "FirstName": "Alberto",  "LastName": "Korda",  "Phone": null  },  ...  ] |

#### Landscape Photographers

Write a program that creates file landscape-photographers.json that should contain the **first name, last name, primary camera make** and **count of their lenses** for all photogaphers whose **primary camera is DSLR** and also they have only **lenses that has focal length no bigger than 30mm** (inclusive). Order them **alphabetically by first name**.

### Example

#### Output

|  |
| --- |
| **landscape-photogaphers.json** |
| [  {  "FirstName": "Allyson",  "LastName": "Gayman",  "CameraMake": "Canon",  "LensesCount": 1  },  ...  ] |

### Exporting to XML

#### Photographers with Same Camera Make

Write a program that creates file same-cameras-photographers.xml that would include information about **all photographers** (**full name** of the photographer; **primary camera make and model**; **list of all their lenses** in the format provided below) whose **primary and the secondary camera are from the same make**. If the photographer do not have any lens **do not add them as part of the XML**.

### Format

<photographers>

<photographer name=”{firstName} {lastName}” primary-camera=”{make} {model}”>

<lenses>

<lens>{Make} {focalLength}mm f{maxAperture}</lens>

...

</lenses>

</photographer>

<photographer name=”{firstName} {lastName}” primary-camera=”{make} {model}” />

...

</photographers>

### Example

#### Output

|  |
| --- |
| **same-cameras-photographers.xml** |
| <?xml version="1.0" encoding="utf-8"?>  <photographers>  ...  <photographer name="Samuel Fosso" primary-camera="Canon 500D">  <lenses>  <lens>Canon 50mm f1.8</lens>  </lenses>  </photographer>  <photographer name="Zhang Xiao" primary-camera="Canon 500D" />  <photographer name="Alberto Korda" primary-camera="Sony A55">  <lenses>  <lens>Tokina 100mm f1.2</lens>  </lenses>  </photographer>  < photographer name="Yan Greenidge" primary-camera="Sony A55">  <lenses>  <lens>Canon 50mm f1.8</lens>  <lens>Nikon 35mm f2</lens>  </lenses>  </photographer>  ...  </photographers> |

#### Workshops by Location

Write a program that creates file workshops-by-location.xml that would include summarized information about **wokshops grouped by location**. Include **only workshops** that have **at least 5 participants**. If a location **does not have any workshops** that have at least 5 participants **skip that location**. Include the **total profit** for each workshop **using the formula**:

Total Profit = ParticipantsCount \* PricePerSingleParticipant – 20% (Trainer Salary)

For each workshop include **list of participants** (their **first** and **last name** separated with single space) and **their count**. The order of locations, workshops and participants does not matter.

**Use the format provided below** for better understanding how your XML output file should look like.

### Format

<locations>

<location name=”{locationName}”>

<workshop name=”{workshopName}”, total-profit=”{totalProfit}”>

<participants count=”{participantsCount}”>

<participant>{firstName} {lastName}</participant>

...

</paricipants>

</workshop>

...

</location>

...

</locations>

### Example

#### Output

|  |
| --- |
| **workshops-by-location.xml** |
| <?xml version="1.0" encoding="utf-8"?>  <locations>  <location name="Sofia">  <workshop name="Street Photography" total-profit="280.000">  <participants count="5">  <participant>Amparo Goers</participant>  <participant>Leon Recio</participant>  <participant>Alica Lacher</participant>  <participant>Bunny Nolan</participant>  <participant>Lyle Longworth</participant>  </participants>  </workshop>  </location>  ...  </locations> |