Washington State University
CPT_S 415 – Big Data
Online

Srinivasulu Badri

Assignment 2

Name: Nam Jun Lee

Student Number: 11606459

1. [**Relational Algebra**] Consider the following database schema:

Movies (Title, Director, Actor);
Location (Theater, Address, Phone number);
Schedule (Theater, Title, Time).

Express the following queries in relational algebra (select σ, project ∏ , Cartesian product X, join (theta-join))

Q1: which theaters feature “Zootopia”?

$$\pi_{\text{Theater}} (\sigma_{\text{Title="Zootopia"}} (\textbf{Schedule}))$$

Q2: List the names and address of theaters featuring a film directed by Steven Spielberg.

$$\pi_{\text{Theater,Address}} (\sigma_{\text{Director="Steven Spielberg"}} (\textbf{Location} \bowtie_{\text{Theater}} \textbf{Schedule} \bowtie_{\text{Title}} \textbf{Movies}))$$

Q3: What are the address and phone number of the Le Champo theater?

$$\pi_{\text{Address,Phone number}} (\sigma_{\text{Theater="Le Champo"}} (\textbf{Location}))$$

Q4: List pairs of actors that acted in the same movie. (* you want to use renaming on Movies and join the Movies with its copy Movie’).

$$\pi_{\textbf{Movies1}.\text{Actor},\textbf{Movies2}.\text{Actor}} (\sigma_{\textbf{Movies1}.\text{Actor} \,!= \textbf{Movies2}.\text{Actor}} (\textbf{Movies1} \bowtie_{\text{Title}} \textbf{Movies2}))$$

2. **[Join Operators]** This sets of questions test the understanding of basic database search operators. Consider a join $\bowtie_{R.A=S.B}$. We ignore the cost of output the result, and measure the cost with the number of I/Os. Given the information about relations to be joined below:

Relation S contains 20,000 tuples and has 10 tuples per block. Relation R contains 100,000 tuples and has 10 tuples per block. Attribute B is the primary key of S. In total, 52 blocks are available in memory. Assume neither relation has any index.

a. Describe a block nested join algorithm, Give the cost of joining R and S with a block nested loops join.

The Block Nested-Loop join algorithm is a nested loop join variant in which all blocks of an internal relation are paired with all blocks of an external relation.

Principles of Operation:

For each block B_s of S

  For each block B_R of R

    For each tuple t_S in S

      For each tuple t_R in R

        Test if pair (t_s, t_R) satisfy the join condition θ

      End for

    End for

  End for

End for

Catalog information for cost estimation before finding the cost:

S = External relation (primary key); R = Internal relation

B_s= 20,000 / 10 = 2,000; B_R=100,000 / 10 = 10,000; Available memory: M = 52

So, Total cost = $(B_s + \frac{B_s}{M-2} \times B_r)$ = 2,000 + (2000/(52-2)) × 10,000

= 2,000 + 40 × 10,000

= **402,000**

b. Describe a sort-merge join algorithm. Give the cost of joining R and S with a sort-merge join.

The Sort Merge join algorithm matches and aligns all pairs with the same value of the join attribute. Then merge and combine these ordered relations. First, created sorted sub-lists. And read M blocks of relation into memory, sort the in-memory blocks as sorted sub-list, and write sorted sub-list to disk repeatedly.

Catalog information for cost estimation before finding the cost:

S = External relation (primary key); R = Internal relation

B_s= 20,000 / 10 = 2,000; B_R=100,000 / 10 = 10,000; Available memory: M = 52

So, in this case (N >= M):

Total cost = $2 \times B_s(1 + \log_{M-1}(B_s/M)) + 2 \times B_R(1 + \log_{M-1}(B_R/M)) + B_s + B_R$

$= 2 \times 2000(1 + \log_{52-1}(2000/52)) + 2 \times 10000(1 + \log_{52-1}(10000/52)) + 2000 + 10000$

$= 2 \times 2000(1 + \log_{51}(38.46153)) + 2 \times 10000(1 + \log_{51}(192.30769)) + 2000 + 10000$

$= 2 \times 2000(1 + 0.92823) + 2 \times 10000(1 + 1.33757) + 2000 + 10000$

$= 2 \times 3856.46 + 2 \times 23375.7 + 2000 + 10000$

$= 66464.32 \approx \mathbf{66,464}$

c. Describe a hash-join algorithm. Give the cost of joining R and S with a hash join.

Hash-join algorithm applicable for equijoins and natural joins. Partition the relation build input and probe input using a hashing function. When partitioning a relation, one block of memory is reserved as the output buffer for each partition.

Catalog information for cost estimation before finding the cost:

S = External relation (primary key); R = Internal relation

B_s= 20,000 / 10 = 2,000; B_R=100,000 / 10 = 10,000; Available memory: M = 52

In this case, not required recursive partitioning:

$3(B_s + B_R) + 4n_h$ block transfers. = approx. $3(B_s + B_R)$

So,

Total cost = 3(2000 + 10000)

= 6000 + 30000

= **36,000**

3. **[XML]**

a. Consider the attached data for a simple Construction Project Management company. Create XML documents (.xml files) to store the data. Provide the XML documents as separate files.

a_employee.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE employees SYSTEM "b_employee.dtd">
<employees>
    <!--employee-->
    <employee emp_num="101">
        <name>
            <emp_lname>News</emp_lname>
            <emp_fname>John</emp_fname>
        </name>
        <emp_initial>G</emp_initial>
        <emp_hiredate>1998-11-08</emp_hiredate>
    </employee>

    <employee emp_num="102">
        <name>
            <emp_lname>Senior</emp_lname>
            <emp_fname>David</emp_fname>
        </name>
        <emp_initial>H</emp_initial>
        <emp_hiredate>1987-07-12</emp_hiredate>
    </employee>

    <employee emp_num="103">
        <name>
            <emp_lname>Arbough</emp_lname>
            <emp_fname>June</emp_fname>
        </name>
        <emp_initial>E</emp_initial>
        <emp_hiredate>1994-12-01</emp_hiredate>
    </employee>

    <employee emp_num="104">
        <name>
            <emp_lname>Ramoras</emp_lname>
            <emp_fname>Anne</emp_fname>
        </name>
        <emp_initial>K</emp_initial>
        <emp_hiredate>1985-11-15</emp_hiredate>
    </employee>

    <employee emp_num="105">
        <name>
            <emp_lname>Johnson</emp_lname>
            <emp_fname>Alice</emp_fname>
        </name>
        <emp_initial>K</emp_initial>
        <emp_hiredate>1991-02-01</emp_hiredate>
    </employee>

    <employee emp_num="106">
        <name>
            <emp_lname>Smithfield</emp_lname>
            <emp_fname>William</emp_fname>
        </name>
        <emp_initial></emp_initial>
        <emp_hiredate>2003-06-22</emp_hiredate>
```

```
    </employee>

    <employee emp_num="112">
        <name>
            <emp_lname>Smithson</emp_lname>
            <emp_fname>Darlene</emp_fname>
        </name>
        <emp_initial>M</emp_initial>
        <emp_hiredate>1992-10-23</emp_hiredate>
    </employee>

    <employee emp_num="114">
        <name>
            <emp_lname>Jones</emp_lname>
            <emp_fname>Annelise</emp_fname>
        </name>
        <emp_initial></emp_initial>
        <emp_hiredate>1991-08-20</emp_hiredate>
    </employee>

    <employee emp_num="118">
        <name>
            <emp_lname>Frommer</emp_lname>
            <emp_fname>James</emp_fname>
        </name>
        <emp_initial>J</emp_initial>
        <emp_hiredate>2004-01-04</emp_hiredate>
    </employee>
</employees>
```

a_project.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE projects SYSTEM "b_project.dtd">
<projects>
    <!--project-->
    <project proj_num="15">
        <proj_name>Evergreen</proj_name>
    </project>

    <project proj_num="18">
        <proj_name>Amber Wave</proj_name>
    </project>
</projects>
```

a_employee_project.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE employeeProjects SYSTEM "b_employee_project.dtd">
<employeeProjects>
    <!--employeeProject-->
    <employeeProject>
        <proj_num>15</proj_num>
        <emp_num>103</emp_num>
        <job_class>Electrical Engineer</job_class>
        <chg_hour>84.50</chg_hour>
        <hours>23.8</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>15</proj_num>
        <emp_num>101</emp_num>
        <job_class>Database Designer</job_class>
        <chg_hour>105.00</chg_hour>
        <hours>19.4</hours>
```

```
    </employeeProject>

    <employeeProject>
        <proj_num>15</proj_num>
        <emp_num>105</emp_num>
        <job_class>Database Designer</job_class>
        <chg_hour>105.00</chg_hour>
        <hours>35.7</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>15</proj_num>
        <emp_num>106</emp_num>
        <job_class>Programmer</job_class>
        <chg_hour>35.75</chg_hour>
        <hours>12.6</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>15</proj_num>
        <emp_num>102</emp_num>
        <job_class>Systems Analyst</job_class>
        <chg_hour>96.75</chg_hour>
        <hours>23.8</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>18</proj_num>
        <emp_num>114</emp_num>
        <job_class>Applications Designer</job_class>
        <chg_hour>48.10</chg_hour>
        <hours>24.6</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>18</proj_num>
        <emp_num>118</emp_num>
        <job_class>General Support</job_class>
        <chg_hour>18.36</chg_hour>
        <hours>45.3</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>18</proj_num>
        <emp_num>104</emp_num>
        <job_class>Systems Analyst</job_class>
        <chg_hour>96.75</chg_hour>
        <hours>32.4</hours>
    </employeeProject>

    <employeeProject>
        <proj_num>18</proj_num>
        <emp_num>112</emp_num>
        <job_class>DSS Analyst</job_class>
        <chg_hour>45.95</chg_hour>
        <hours>44.0</hours>
    </employeeProject>
</employeeProjects>
```

b. Consider the XML documents defined in (a) above for the Construction Project Management dataset. Give a Document Type Definition (DTD) representation for each of the XML documents. Provide the DTD representations as .dtd files. How do you encode keys? Foreign keys?

To do key encoding <!ATTLIST attribute_name ID> has been applied and the foreign key is <!ATTLIST attribute_name IDREF> was applied.

b_project.dtd:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE projects[
    <!ELEMENT projects(project*)>
    <!ELEMENT project (proj_num, proj_name)>
    <!ELEMENT proj_num (#PCDATA)>
    <!ELEMENT proj_num (#PCDATA)>
    <!ATTLIST project proj_num ID #required>
]>
```

b_employee.dtd:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE employees[
    <!ELEMENT employees(employee*)>
    <!ELEMENT employee (emp_num, name+, emp_initial, emp_hiredate)>
    <!ELEMENT emp_num (#PCDATA)>
    <!ELEMENT name (emp_lname, emp_fname)
    <!ELEMENT emp_initial (#PCDATA)>
    <!ELEMENT emp_hiredate (#PCDATA)>
    <!ELEMENT emp_lname (#PCDATA)>
    <!ELEMENT emp_fname (#PCDATA)>
    <!ATTLIST employee emp_num ID #required>
    <!ATTLIST employee emp_initial EMPTY #IMPLIED>
]>
```

b_employee_project.dtd:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE employeeProjects[
    <!ELEMENT employeeProjects(employeeProject*)>
    <!ELEMENT employeeProject(proj_num, emp_num, job_class, chg_hour, hours)>
```

```
    <!ELEMENT proj_num (#PCDATA)>
    <!ELEMENT emp_num (#PCDATA)>
    <!ELEMENT job_class (#PCDATA)>
    <!ELEMENT chg_hour (#PCDATA)>
    <!ELEMENT hours (#PCDATA)>
    <!ATTLIST employeeProject proj_num IDREF #required>
    <!ATTLIST employeeProject emp_num IDREF #required>
]>
```

c. Consider the XML documents defined in (a) above for the Construction Project Management dataset. Give an XML schema representation for each of the XML documents. Provide the XML schema representation as .xsd files. How do you encode keys? Foreign keys?

To do key encoding:

<xs:key name=’’ >

<xs:selector xpath=’’ />

<xs:field xpath=’’ />

</xs:key>

And the foreign key:

<xs:keyref name=’’, refer=’’>

<xs:selector xpath=’’ />

<xs:field xpath=’’ />

</xs:keyref>

c_project.xsd:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://codingsam.com"
    xmlns="http://codingsam.com"
    elementFormDefault="qualified">
    <xs:element name="projects">
        <xs:complexType>
            <xs:sequence>
                <xs:element maxOccurs="unbounded" name="project">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="proj_name" type="
                            xs:string" />
                        </xs:sequence>
                        <xs:attribute name="proj_num" type="xs:string"
                         use="required" />
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>

        <xs:key name="proj_num_key" >
            <xs:selector xpath=".//project" />
            <xs:field xpath="proj_num" />
        </xs:key>

    </xs:element>
</xs:schema>
```

c_employee.xsd:

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xs:element name="employees">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="employee">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="emp_lname" type="xs:string" />
                    <xs:element name="emp_fname" type="xs:string" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="emp_initial" type="xs:string" nillable
              ="true" />
              <xs:element name="emp_hiredate" type="xs:date" />
            </xs:sequence>
            <xs:attribute name="emp_num" type="xs:string" use="
            required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>

    <xs:key name="emp_num_key" >
      <xs:selector xpath=".//employee" />
      <xs:field xpath = "emp_num" />
    </xs:key>

  </xs:element>
</xs:schema>
```

c_employee_project.xsd:

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://codingsam.com"
    xmlns="http://codingsam.com"
    elementFormDefault="qualified">
  <xs:element name="employeeProjects">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="employeeProject">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="proj_num" type="xs:string" />
              <xs:element name="emp_num" type="xs:string" />
              <xs:element name="job_class" type="xs:string" />
              <xs:element name="chg_hour" type="xs:decimal" />
              <xs:element name="hours" type="xs:decimal" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>

    <xs:keyref name="proj_num_ref" refer="proj_num_key">
      <xs:selector xpath=".//employeeProject" />
      <xs:field xpath="proj_num" />
    </xs:keyref>

    <xs:keyref name="emp_num_ref" refer="emp_num_key">
      <xs:selector xpath=".//employeeProject" />
      <xs:field xpath="emp_num" />
    </xs:keyref>

  </xs:element>
</xs:schema>
```

4. **[JSON]**

a. Consider again the Construction Project Management company dataset attached with this assignment. Assuming you don't have any relational schema. How would you model the data to store it as JSON documents. Use the provided data to create the JSON documents. Provide the JSON documents as .json files.

Model the data to store as JSON:

Data objects defined within {}; Data objects consists of a collection of key – value pairs.

Data model for nested JSON to use.

a_project.json:

```
[
{
    "PROJ_NUM": 15,
    "PROJ_NAME": "Evergreen"
},
{
    "PROJ_NUM": 18,
    "PROJ_NAME": "Amber Wave"
}
]
```

a_employee.json:


```
[
{
    "EMP_NUM": 101,
    "EMP_LNAME": "News",
    "EMP_FNAME": "John",
    "EMP_INITIAL": "G",
    "EMP_HIREDATE": "11/08/1998"
},
{
    "EMP_NUM": 102,
    "EMP_LNAME": "Senior",
    "EMP_FNAME": "David",
    "EMP_INITIAL": "H",
    "EMP_HIREDATE": "07/12/1987"
},
{
    "EMP_NUM": 103,
    "EMP_LNAME": "Arbough",
    "EMP_FNAME": "June",
    "EMP_INITIAL": "E",
    "EMP_HIREDATE": "12/01/1994"
},
{
    "EMP_NUM": 104,
    "EMP_LNAME": "Ramoras",
    "EMP_FNAME": "Anne",
    "EMP_INITIAL": "K",
    "EMP_HIREDATE": "11/15/1985"
},
{
    "EMP_NUM": 105,
    "EMP_LNAME": "Johnson",
    "EMP_FNAME": "Alice",
    "EMP_INITIAL": "K",
    "EMP_HIREDATE": "02/01/1991"
},
{
    "EMP_NUM": 106,
    "EMP_LNAME": "Smithfield",
    "EMP_FNAME": "William",
    "EMP_INITIAL": "",
    "EMP_HIREDATE": "06/22/2003"
},
{
    "EMP_NUM": 112,
    "EMP_LNAME": "Smithson",
    "EMP_FNAME": "Darlene",
    "EMP_INITIAL": "M",
    "EMP_HIREDATE": "10/23/1992"
},
{
    "EMP_NUM": 114,
    "EMP_LNAME": "Jones",
    "EMP_FNAME": "Annelise",
    "EMP_INITIAL": "",
    "EMP_HIREDATE": "08/20/1991"
},
{
    "EMP_NUM": 118,
    "EMP_LNAME": "Frommer",
    "EMP_FNAME": "James",
    "EMP_INITIAL": "J",
```

```
    "EMP_HIREDATE": "01/04/2004"
}
]
```


a_employee_project.json:


```
[
{
    "PROJ_NUM": 15,
    "EMP_NUM": 103,
    "JOB_CLASS": "Electrical Engineer",
    "CHG_HOUR": 84.50,
    "HOURS": 23.8
},
{
    "PROJ_NUM": 15,
    "EMP_NUM": 101,
    "JOB_CLASS": "Database Designer",
    "CHG_HOUR": 105.00,
    "HOURS": 19.4
},
{
    "PROJ_NUM": 15,
    "EMP_NUM": 105,
    "JOB_CLASS": "Database Designer",
    "CHG_HOUR": 105.00,
    "HOURS": 35.7
},
{
    "PROJ_NUM": 15,
    "EMP_NUM": 106,
    "JOB_CLASS": "Programmer",
    "CHG_HOUR": 35.75,
    "HOURS": 12.6
},
{
    "PROJ_NUM": 15,
    "EMP_NUM": 102,
    "JOB_CLASS": "Systems Analyst",
    "CHG_HOUR": 96.75,
    "HOURS": 23.8
},
{
    "PROJ_NUM": 18,
    "EMP_NUM": 114,
    "JOB_CLASS": "Applications Designer",
    "CHG_HOUR": 48.10,
    "HOURS": 24.6
},
{
    "PROJ_NUM": 18,
    "EMP_NUM": 118,
    "JOB_CLASS": "General Support",
    "CHG_HOUR": 18.36,
    "HOURS": 45.3
},
{
    "PROJ_NUM": 18,
    "EMP_NUM": 104,
    "JOB_CLASS": "Systems Analyst",
    "CHG_HOUR": 96.75,
    "HOURS": 32.4
},
```

```
{
    "PROJ_NUM": 18,
    "EMP_NUM": 112,
    "JOB_CLASS": "DSS Analyst",
    "CHG_HOUR": 45.95,
    "HOURS": 44.0
}
]
```


a_data_model.json:


```
[
{
    "PROJ_NUM": 15,
    "PROJ_NAME": "Evergreen",
    "EMPLOYEE": [
        {
            "EMP_NUM": 101,
            "EMP_LNAME": "News",
            "EMP_FNAME": "John",
            "EMP_INITIAL": "G",
            "EMP_HIREDATE": "11/08/1998"
        }
    ],
    "JOB_CLASS": "Database Designer",
    "CHG_HOUR": 105.00,
    "HOURS": 19.4
},
{
    "PROJ_NUM": 15,
    "PROJ_NAME": "Evergreen",
    "EMPLOYEE": [
        {
            "EMP_NUM": 102,
            "EMP_LNAME": "Senior",
            "EMP_FNAME": "David",
            "EMP_INITIAL": "H",
            "EMP_HIREDATE": "07/12/1987"
        }
    ],
    "JOB_CLASS": "Systems Analyst",
    "CHG_HOUR": 96.75,
    "HOURS": 23.8
},
{
    "PROJ_NUM": 15,
    "PROJ_NAME": "Evergreen",
    "EMPLOYEE": [
        {
            "EMP_NUM": 103,
            "EMP_LNAME": "Arbough",
            "EMP_FNAME": "June",
            "EMP_INITIAL": "E",
            "EMP_HIREDATE": "12/01/1994"
        }
    ],
    "JOB_CLASS": "Electrical Engineer",
    "CHG_HOUR": 84.50,
    "HOURS": 23.8
},
{
    "PROJ_NUM": 15,
    "PROJ_NAME": "Evergreen",
```

```
        "EMPLOYEE": [
            {
                "EMP_NUM": 105,
                "EMP_LNAME": "Johnson",
                "EMP_FNAME": "Alice",
                "EMP_INITIAL": "K",
                "EMP_HIREDATE": "02/01/1991"
            }
        ],
        "JOB_CLASS": "Database Designer",
        "CHG_HOUR": 105.00,
        "HOURS": 35.7
    },
    {
        "PROJ_NUM": 15,
        "PROJ_NAME": "Evergreen",
        "EMPLOYEE": [
            {
                "EMP_NUM": 106,
                "EMP_LNAME": "Smithfield",
                "EMP_FNAME": "William",
                "EMP_INITIAL": "",
                "EMP_HIREDATE": "06/22/2003"
            }
        ],
        "JOB_CLASS": "Programmer",
        "CHG_HOUR": 35.75,
        "HOURS": 12.6
    },

    {
        "PROJ_NUM": 18,
        "PROJ_NAME": "Amber Wave",
        "EMPLOYEE": [
            {
                "EMP_NUM": 104,
                "EMP_LNAME": "Ramoras",
                "EMP_FNAME": "Anne",
                "EMP_INITIAL": "K",
                "EMP_HIREDATE": "11/15/1985"
            }
        ],
        "JOB_CLASS": "Systems Analyst",
        "CHG_HOUR": 96.75,
        "HOURS": 32.4
    },
    {
        "PROJ_NUM": 18,
        "PROJ_NAME": "Amber Wave",
        "EMPLOYEE": [
            {
                "EMP_NUM": 112,
                "EMP_LNAME": "Smithson",
                "EMP_FNAME": "Darlene",
                "EMP_INITIAL": "M",
                "EMP_HIREDATE": "10/23/1992"
            }
        ],
        "JOB_CLASS": "DSS Anaylst",
        "CHG_HOUR": 45.95,
        "HOURS": 44.0
    },
    {
        "PROJ_NUM": 18,
        "PROJ_NAME": "Amber Wave",
        "EMPLOYEE": [
            {
```

```
                "EMP_NUM": 114,
                "EMP_LNAME": "Jones",
                "EMP_FNAME": "Annelise",
                "EMP_INITIAL": "",
                "EMP_HIREDATE": "08/20/1991"
            }
        ],
        "JOB_CLASS": "Applicatioins Designer",
        "CHG_HOUR": 48.10,
        "HOURS": 24.6
},
{
        "PROJ_NUM": 18,
        "PROJ_NAME": "Amber Wave",
        "EMPLOYEE": [
            {
                "EMP_NUM": 118,
                "EMP_LNAME": "Frommer",
                "EMP_FNAME": "James",
                "EMP_INITIAL": "J",
                "EMP_HIREDATE": "01/04/2004"
            }
        ],
        "JOB_CLASS": "General Support",
        "CHG_HOUR": 18.36,
        "HOURS": 45.3
}
]
```


b. Consider the JSON data model and JSON documents defined in (a) for the Construction Project Management dataset. Give a JSON Schema that can be used to validate the JSON documents with sample data. Provide the JSON Schema as .json files.

b_project.json:


```
{
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "$id": "https://wsu.edu/cpts415/schemas/a_project.json",
    "type": "object",
    "properties":
    {
        "PROJ_NUM": {"type": "string"},
        "PROJ_NAME": {"type": "string"}
    },
    "required": ["PROJ_NUM"]
}
```


b_employee.json:


```
{
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "$id": "https://wsu.edu/cpts415/schemas/a_employee.json",
    "type": "object",
    "properties":
    {
        "EMP_NUM": {"type": "string"},
```

```
            "EMP_LNAME": {"type": "string"},
            "EMP_FNAME": {"type": "string"},
            "EMP_INITIAL": {"type": ["string", "null"]},
            "EMP_HIREDATE": {"type": "string", "format": "date"}
        },
        "required": ["EMP_NUM"]
}
```

b_employee_project.json:

```
{
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "$id": "https://wsu.edu/cpts415/schemas/a_employee_project.json",
    "type": "object",
    "properties":
    {
        "PROJ_NUM": {"type": "string"},
        "EMP_NUM": {"type": "string"},
        "JOB_CLASS": {"type": "string"},
        "CHG_HOUR": {"type": "number"},
        "HOURS": {"type": "number"}
    },
    "required": ["PROJ_NUM","EMP_NUM"]
}
```

b_data_model.json:

```
{
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "$id": "https://wsu.edu/cpts415/schemas/a_data_model.json",
    "type": "object",
    "properties":
    {
        "PROJ_NUM": {"type": "string"},
        "PROJ_NAME": {"type": "string"},
        "EMPLOYEE":
        {
            "type": "array",
            "items":
            {
                "type": "object",
                "properties":
                {
                    "EMP_NUM": {"type": "string"},
                    "EMP_LNAME": {"type": "string"},
                    "EMP_FNAME": {"type": "string"},
                    "EMP_INITIAL": {"type": ["string", "null"]},
                    "EMP_HIREDATE": {"type": "string", "format": "date"}
                }
            }
        },
        "JOB_CLASS": {"type": "string"},
        "CHG_HOUR": {"type": "number"},
        "HOURS": {"type": "number"}
    },
    "required": ["PROJ_NUM", "EMP_NUM"]
}
```