**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”?

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

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

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’).

2. **[Join Operators]** This sets of questions test the understanding of basic database search operators. Consider a join ⋈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.

1. 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 of S

For each block of R

For each tuple in S

For each tuple in R

Test if pair ( satisfy the join condition

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

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

So, Total cost = ( + ) = 2,000 + (2000/(52-2)) 10,000

= 2,000 + 40 10,000

= **402,000**

1. 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

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

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

Total cost =

=

=

=

=

= 66464.32

1. 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

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

In this case, not required recursive partitioning:

3( + ) + 4 block transfers. = approx. 3( + )

So,

Total cost = 3( 2000 + 10000)

= 6000 + 30000

= **36,000**

3. **[XML]**

1. 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>**

1. 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>

]>

1. 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=’’ />

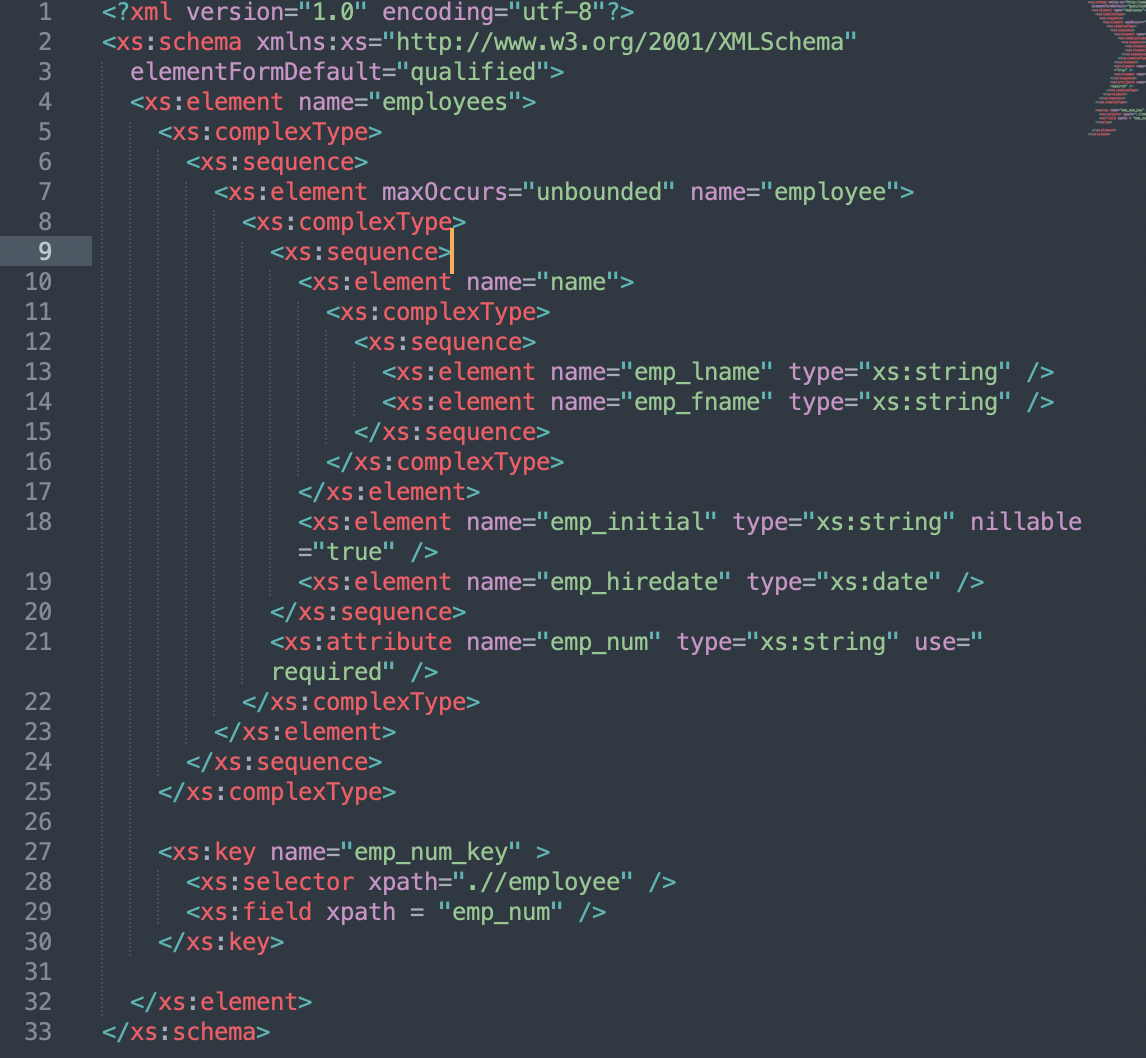
</xs:keyref>

c\_project.xsd:

텍스트이(가) 표시된 사진

자동 생성된 설명

c\_employee.xsd:



c\_employee\_project.xsd:



4. **[JSON]**

1. 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

}

]

1. 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"]

}