CS549 Coursework

by:

Group number		XX	
Student ID	**Student email**	**Student name**	**Student's signature**
202392175	**vishesh.kishore.2023@uni.strath.ac.uk**	**Vishesh Kishore**	Vishesh Kishore
202456426	**ali.nasir.2023@uni.strath.ac.uk**	**Ali Nasir**	Ali Nasir
202462898	**andrew.kiggins.2023@uni.strath.ac.uk**	**Andrew Kiggins**	
	gabriel.agbese.2023@uni.strath.ac.uk	**Gabriel Agbese**	

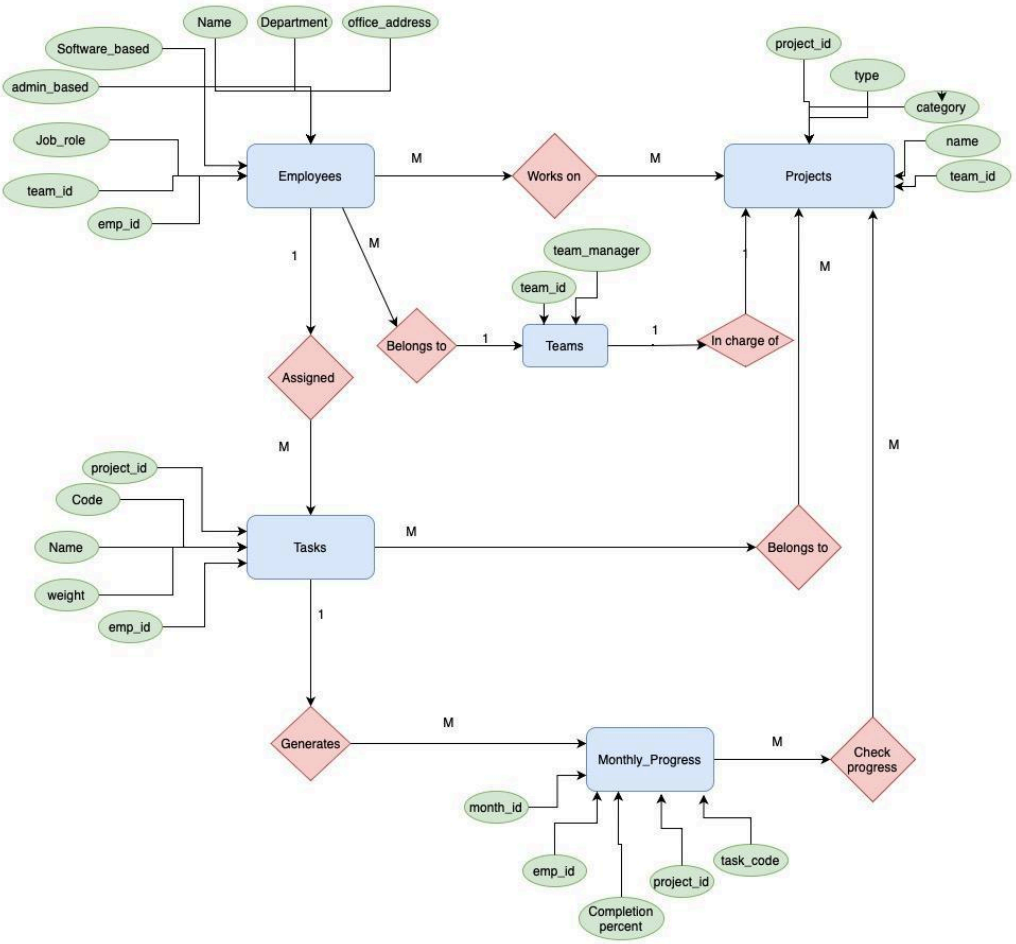
Department of
Computer and Information Sciences

March 2024

Contents

1 Relational database

1.1 Entity relationship diagram



1.2 SQL statements to implement ERD

```
-- Employees
create table Employees(
    emp_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    Name varchar(255) NOT NULL,
    department varchar(255) NOT NULL,
    software_based boolean,
    admin_based boolean,
    job_role varchar(255),
    team_id INT,
    office_location varchar(255),
    CONSTRAINT CHK_based CHECK (software_based OR admin_based)
);


-- Teams
create table Teams(
    team_id INT NOT NULL PRIMARY KEY,
    team_manager INT
);

-- Adding foreign key constraints in Employee and Teams table
```

```sql
ALTER TABLE Employees ADD FOREIGN KEY (team_id) REFERENCES Teams(team_id);
ALTER TABLE Teams ADD FOREIGN KEY (team_manager) REFERENCES Employees(emp_id);

-- Projects
create table Projects(
   project_id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   name varchar(255) NOT NULL,
   type varchar(255) NOT NULL,
   category varchar(255) NOT NULL,
   team_id INT,
   project_manager INT,
   progress FLOAT CHECK (progress BETWEEN 0.0 AND 1.0),
   FOREIGN KEY (team_id) REFERENCES Teams(team_id)
);

-- Tasks
create table Tasks(
   Code INT NOT NULL,
   name varchar(255),
   weight INT CHECK (weight Between 1 and 5),
   project_id INT NOT NULL,
   emp_id INT NOT NULL,
   PRIMARY KEY (Code, project_id, emp_id),
   FOREIGN KEY (project_id) REFERENCES Projects(project_id),
   FOREIGN KEY (emp_id) REFERENCES Employees(emp_id)
);

-- Progress

create table Monthly_Progress(
   month_id INT NOT NULL,
   emp_id INT NOT NULL,
   project_id INT NOT NULL,
   task_code INT NOT NULL,
   completion_percent INT CHECK (completion_percent BETWEEN 0 AND 100),
   PRIMARY KEY (project_id, task_code),
   FOREIGN KEY (emp_id) REFERENCES Employees(emp_id),
   FOREIGN KEY (project_id) REFERENCES Projects(project_id),
   FOREIGN KEY (task_code) REFERENCES Tasks(Code)
);
```

1.3 SQL statements to insert data

```sql
-- Insert values in Employees Table

insert into Employees values(1, "John Doe", "Engineering", 1, 0, "Software Developer", 111,
"London"),
(2, "Jane Smith", "Marketing", 0, 1, "Marketing Specialist", 222 , "Glasgow"),
(3, "Michael Johnson", "Finance", 0, 1, "Accountant", 333, "Manchester"),
(4, "Emily Brown", "HR", 0, 1, "HR Manager ", 333, "Edinburgh"),
(5, "Chris Lee", "Engineering", 1, 1, "Software Developer", 111, "Edinburgh");

-- Insert values in Teams table

insert into Teams values(111, 1),
```

```
(222, 2),
(333, 4);

-- Insert values in Projects

insert into Projects values(1, "InnovateX", "Web App", "Enterprise Software", 111, 1, .7),
(2, "Salesforce CRM", "Desktop App", "Enterprise Software", 111, 1, .06),
(3, "ESS Portal", "Desktop App", "Payrol Software", 333, 1, .5);


-- Insert values in Tasks

insert into Tasks values(101, "Frontend Design", 4, 1, 1),
(101,"Frontend Design",4 ,2, 5),
(102, "Frontend Testing", 3, 1, 5);


-- Insert values in Monthly_Progress

insert into Monthly_Progress values(1, 2, 1, 101, 100),
(2, 1, 3, 101, 50),
(3, 5, 1, 102, 10);
```

1.4 Tables with data shown.

```
MariaDB [test]> select * from Employees;
+--------+----------------+-------------+---------------+-------------+---------------------+---------+-----------------+
| emp_id | Name           | department  | software_based | admin_based | job_role            | team_id | office_location |
+--------+----------------+-------------+---------------+-------------+---------------------+---------+-----------------+
|      1 | John Doe       | Engineering |             1 |           0 | Software Developer  |     111 | London          |
|      2 | Jane Smith     | Marketing   |             0 |           1 | Marketing Specialist |     222 | Glasgow         |
|      3 | Michael Johnson | Finance     |             0 |           1 | Accountant          |     333 | Manchester      |
|      4 | Emily Brown    | HR          |             0 |           1 | HR Manager          |     333 | Edinburgh       |
|      5 | Chris Lee      | Engineering |             1 |           1 | Software Developer  |     111 | Edinburgh       |
+--------+----------------+-------------+---------------+-------------+---------------------+---------+-----------------+
5 rows in set (0.019 sec)
```

```
MariaDB [test]> select * from Projects;
+------------+----------------+-------------+---------------------+---------+-----------------+----------+
| project_id | name           | type        | category            | team_id | project_manager | progress |
+------------+----------------+-------------+---------------------+---------+-----------------+----------+
|          1 | InnovateX      | Web App     | Enterprise Software |     111 |               1 |      0.7 |
|          2 | Salesforce CRM | Desktop App | Enterprise Software |     111 |               1 |     0.06 |
|          3 | ESS Portal     | Desktop App | Payrol Software     |     333 |               1 |      0.5 |
+------------+----------------+-------------+---------------------+---------+-----------------+----------+
3 rows in set (0.007 sec)
```

```
MariaDB [test]> select * from Tasks;
+-------+------------------+--------+------------+--------+
| Code  | name             | weight | project_id | emp_id |
+-------+------------------+--------+------------+--------+
|  101  | Frontend Design  |      4 |          1 |      1 |
|  101  | Frontend Design  |      4 |          2 |      5 |
|  102  | Frontend Testing |      3 |          1 |      5 |
+-------+------------------+--------+------------+--------+
3 rows in set (0.003 sec)
```

```
MariaDB [test]> select * from Teams;
+---------+--------------+
| team_id | team_manager |
+---------+--------------+
|     111 |            1 |
|     222 |            2 |
|     333 |            4 |
+---------+--------------+
3 rows in set (0.009 sec)
```

```
MariaDB [test]> select * from Monthly_Progress;
+----------+--------+------------+-----------+--------------------+
| month_id | emp_id | project_id | task_code | completion_percent |
+----------+--------+------------+-----------+--------------------+
|        1 |      2 |          1 |       101 |                100 |
|        3 |      5 |          1 |       102 |                 10 |
|        2 |      1 |          3 |       101 |                 50 |
+----------+--------+------------+-----------+--------------------+
3 rows in set (0.005 sec)
```

1.5 Five SQL queries with results

-- 1. List the ID and name of all employees.
select emp_id, name from Employees;

```
MariaDB [test]> select emp_id, name from Employees;
+--------+-----------------+
| emp_id | name            |
+--------+-----------------+
|      1 | John Doe        |
|      2 | Jane Smith      |
|      3 | Michael Johnson |
|      4 | Emily Brown     |
|      5 | Chris Lee       |
+--------+-----------------+
5 rows in set (0.004 sec)
```

-- 2. List the name of all projects together with their type.
select distinct(Name), type from Projects;

```
MariaDB [test]> select distinct(Name), type from Projects;
+----------------+-------------+
| Name           | type        |
+----------------+-------------+
| InnovateX      | Web App     |
| Salesforce CRM | Desktop App |
| ESS Portal     | Desktop App |
+----------------+-------------+
3 rows in set (0.007 sec)
```

-- 3. List the ID and name of all employees together with the project name and project category that they work on.
select Employees.emp_id, Employees.name, Projects.Name, Projects.Category from Employees, Projects where Employees.team_id = Projects.team_id;

```
MariaDB [test]> select Employees.emp_id, Employees.name, Projects.Name, Projects.Category from Employees, Projects where Employees.team_id = Projects.team_id;
+--------+----------------+---------------+--------------------+
| emp_id | name           | Name          | Category           |
+--------+----------------+---------------+--------------------+
|      1 | John Doe       | InnovateX     | Enterprise Software |
|      1 | John Doe       | Salesforce CRM | Enterprise Software |
|      3 | Michael Johnson | ESS Portal    | Payrol Software    |
|      4 | Emily Brown    | ESS Portal    | Payrol Software    |
|      5 | Chris Lee      | InnovateX     | Enterprise Software |
|      5 | Chris Lee      | Salesforce CRM | Enterprise Software |
+--------+----------------+---------------+--------------------+
6 rows in set (0.005 sec)
```

-- 4. List the ID and name of all tasks together with the name and ID of the employee who works on this task.
select Tasks.Code, Tasks.Name, Employees.emp_id, Employees.Name from Tasks, Employees where Tasks.emp_id = Employees.emp_id;

```
MariaDB [test]> select Tasks.Code, Tasks.Name, Employees.emp_id, Employees.Name from Tasks, Employees where Tasks.emp_id = Employees.emp_id;
+------+----------------+--------+-----------+
| Code | Name           | emp_id | Name      |
+------+----------------+--------+-----------+
|  101 | Frontend Design |      1 | John Doe  |
|  101 | Frontend Design |      5 | Chris Lee |
|  102 | Frontend Testing |      5 | Chris Lee |
+------+----------------+--------+-----------+
3 rows in set (0.008 sec)
```

-- 5. List the ID and name of all employees together with the office number that they are based in.
select emp_id, name, office_location from Employees;

```
MariaDB [test]> select emp_id, name, office_location from Employees;
+--------+----------------+----------------+
| emp_id | name           | office_location |
+--------+----------------+----------------+
|      1 | John Doe       | London         |
|      2 | Jane Smith     | Glasgow        |
|      3 | Michael Johnson | Manchester     |
|      4 | Emily Brown    | Edinburgh      |
|      5 | Chris Lee      | Edinburgh      |
+--------+----------------+----------------+
5 rows in set (0.005 sec)
```

2 XML data model

2.1 XML tree drawing

```
- company
  |
  |-- employees
  |   |
  |   |-- employee
  |   |   |
  |   |   |-- emp_id: 1
  |   |   |-- name: John Doe
  |   |   |-- department: Engineering
  |   |   |-- software_based: false
  |   |   |-- admin_based: true
  |   |   |-- job_role: Software Developer
  |   |   |-- team_id: 111
  |   |   |-- office_location: London
  |   |
  |   |-- employee
  |   |   |
  |   |   |-- emp_id: 2
  |   |   |-- name: Jane Smith
  |   |   |-- department: Marketing
  |   |   |-- software_based: true
  |   |   |-- admin_based: false
  |   |   |-- job_role: Marketing Specialist
  |   |   |-- team_id: 222
  |   |   |-- office_location: Glasgow
  |   |
  |   |-- employee
  |   |   |
  |   |   |-- emp_id: 3
  |   |   |-- name: Michael Johnson
  |   |   |-- department: Finance
  |   |   |-- software_based: false
  |   |   |-- admin_based: true
  |   |   |-- job_role: Accountant
  |   |   |-- team_id: 333
  |   |   |-- office_location: Manchester
  |   |
  |   |-- employee
  |   |   |
  |   |   |-- emp_id: 4
  |   |   |-- name: Emily Brown
  |   |   |-- department: HR
  |   |   |-- software_based: false
  |   |   |-- admin_based: true
  |   |   |-- job_role: HR Manager
  |   |   |-- team_id: 333
```

```
|   |   |-- office_location: Edinburgh
|   |
|   |-- employee
|         |
|         |-- emp_id: 5
|         |-- name: Chris Lee
|         |-- department: Engineering
|         |-- software_based: true
|         |-- admin_based: true
|         |-- job_role: Software Developer
|         |-- team_id: 111
|         |-- office_location: Edinburgh
|
|-- teams
|   |
|   |-- team
|   |   |
|   |   |-- team_id: 111
|   |   |-- manager_id: 1
|   |
|   |-- team
|   |   |
|   |   |-- team_id: 222
|   |   |-- manager_id: 2
|   |
|   |-- team
|         |
|         |-- team_id: 333
|         |-- manager_id: 4
|
|-- projects
|   |
|   |-- project
|   |   |
|   |   |-- project_id: 1
|   |   |-- name: InnovateX
|   |   |-- type: Web App
|   |   |-- category: Enterprise Software
|   |   |-- team_id: 111
|   |   |-- project_manager: 1
|   |   |-- progress: 0.75
|   |
|   |-- project
|   |   |
|   |   |-- project_id: 2
|   |   |-- name: Salesforce CRM
|   |   |-- type: Desktop App
|   |   |-- category: Enterprise Software
|   |   |-- team_id: 111
|   |   |-- project_manager: 1
```

```
|   |   |-- progress: 0.0
|   |
|   |-- project
|         |
|         |-- project_id: 3
|         |-- name: ESS Portal
|         |-- type: Desktop App
|         |-- category: Payroll Software
|         |-- team_id: 333
|         |-- project_manager: 4
|         |-- progress: 0.0
|
|-- tasks
|   |
|   |-- task
|   |   |
|   |   |-- task_id: 101
|   |   |-- name: Frontend Design
|   |   |-- weight: 4
|   |   |-- project_id: 1
|   |   |-- emp_id: 1
|   |
|   |-- task
|   |   |
|   |   |-- task_id: 101
|   |   |-- name: Frontend Design
|   |   |-- weight: 4
|   |   |-- project_id: 2
|   |   |-- emp_id: 5
|   |
|   |-- task
|         |
|         |-- task_id: 102
|         |-- name: Frontend Testing
|         |-- weight: 3
|         |-- project_id: 1
|         |-- emp_id: 5
|
|-- monthly_reports
          |
          |-- monthly_report
          |   |
          |   |-- report_month_id: 1
          |   |-- task_id: 101
          |   |-- emp_id: 2
          |   |-- proj_id: 1
          |   |-- completion_percent: 100
          |
          |-- monthly_report
          |   |
```

```
|   |-- report_month_id: 2
|   |-- task_id: 101
|   |-- emp_id: 1
|   |-- proj_id: 2
|   |-- completion_percent: 50
|
|-- monthly_report
|
|-- report_month_id: 3
|-- task_id: 102
|-- emp_id: 5
|-- proj_id: 1
|-- completion_percent: 10
```

2.2 XML DTD with explanation

```
<!DOCTYPE company [
	<!ELEMENT company (employees, teams, projects, tasks, monthly_reports)>
	<!ELEMENT employees (employee+)>
	<!ELEMENT employee (emp_id, name, department, software_based, admin_based, job_role, team_id,
office_location)>
	<!ELEMENT emp_id (#PCDATA)>
	<!ELEMENT name (#PCDATA)>
	<!ELEMENT department (#PCDATA)>
	<!ELEMENT software_based (#PCDATA)>
	<!ELEMENT admin_based (#PCDATA)>
	<!ELEMENT job_role (#PCDATA)>
	<!ELEMENT team_id (#PCDATA)>
	<!ELEMENT office_location (#PCDATA)>

	<!ELEMENT teams (team+)>
	<!ELEMENT team (team_id, manager_id)>
	<!ELEMENT team_id (#PCDATA)>
	<!ELEMENT manager_id (#PCDATA)>

	<!ELEMENT projects (project+)>
	<!ELEMENT project (project_id, name, type, category, team_id, project_manager, progress)>
	<!ELEMENT project_id (#PCDATA)>
	<!ELEMENT name (#PCDATA)>
	<!ELEMENT type (#PCDATA)>
	<!ELEMENT category (#PCDATA)>
	<!ELEMENT team_id (#PCDATA)>
	<!ELEMENT project_manager (#PCDATA)>
	<!ELEMENT progress (#PCDATA)>

	<!ELEMENT tasks (task+)>
	<!ELEMENT task (task_id, name, weight, project_id, emp_id)>
	<!ELEMENT task_id (#PCDATA)>
	<!ELEMENT name (#PCDATA)>
	<!ELEMENT weight (#PCDATA)>
	<!ELEMENT project_id (#PCDATA)>
	<!ELEMENT emp_id (#PCDATA)>

	<!ELEMENT monthly_reports (monthly_report+)>
	<!ELEMENT monthly_report (report_month_id, task_id, emp_id, proj_id, completion_percent)>
```

```
        <!ELEMENT report_month_id (#PCDATA)>
        <!ELEMENT task_id (#PCDATA)>
        <!ELEMENT emp_id (#PCDATA)>
        <!ELEMENT proj_id (#PCDATA)>
        <!ELEMENT completion_percent (#PCDATA)>
]>
```

The XML DTD that we came up with is a representation of the SQL tables. all of the parent nodes themselves are supposed to represent a table each. This can be seen in the children of the parent nodes. We included the “one or more constraint” to act as a logic check to ensure that there is at least one or more entry for each table. PCDATA was chosen to be more versatile, as the data could then be expressed in other markup formats making the data more extendable.

2.3 Five XQuery queries with results

1. List the ID and name of all employees.

```
for $employee in /company/employees/employee
return
   <employee>
      <employee_id>{$employee/emp_id}</employee_id>
      <employee_name>{$employee/name}</employee_name>
   </employee>
```

Result of the above expression applied to the above input file:

```
1
2          1
3          John Doe
4
5
6          2
7          Jane Smith
8
9
10         3
11         Michael Johnson
12
13
14         4
15         Emily Brown
16
17
18         5
19         Chris Lee
20
21
```

2. List the name of all projects together with their type.

```
for $project in /company/projects/project
return
   <project>
      <project_name>{$project/name}</project_name>
      <project_type>{$project/type}</project_type>
```

</project>

Result of the above expression applied to the above input file:

```

        InnovateX
        Web App

        Salesforce CRM
        Desktop App


        ESS Portal
        Desktop App
```

3. List the ID and name of all employees together with the project name and project category that they work on.

```
for $employee in /company/employees/employee
let $employee_id := $employee/emp_id
for $project in /company/projects/project[team_id = $employee/team_id]
return
    <employee_project>
    <employee_id>{$employee_id}</employee_id>
    <employee_name>{$employee/name}</employee_name>
    <project_name>{$project/name}</project_name>
    <project_category>{$project/category}</project_category>
    </employee_project>
```

Result of the above expression applied to the above input file:

```

        1
        John Doe
        InnovateX
        Enterprise Software


        1
        John Doe
        Salesforce CRM
        Enterprise Software


        3
        Michael Johnson
        ESS Portal
        Payroll Software


        4
        Emily Brown
        ESS Portal
        Payroll Software


        5
        Chris Lee
        InnovateX
        Enterprise Software


        5
        Chris Lee
        Salesforce CRM
        Enterprise Software
```

4. List the ID and name of all tasks together with the name and ID of the employee who works on this task.

```
for $task in /company/tasks/task
return
   <tasks>
      <task_id>{$task/task_id}</task_id>
      <task_name>{$task/name}</task_name>
      <employee_id>{$task/emp_id}</employee_id>
      <employee_name>{/company/employees/employee[emp_id =
$task/emp_id]/name}</employee_name>
   </tasks>
```

Result of the above expression applied to the above input file:

```
        101
        Frontend Design
        1
        John Doe


        101
        Frontend Design
        5
        Chris Lee


        102
        Frontend Testing
        5
        Chris Lee
```

5. List the ID and name of all employees together with the office number that they are based in.

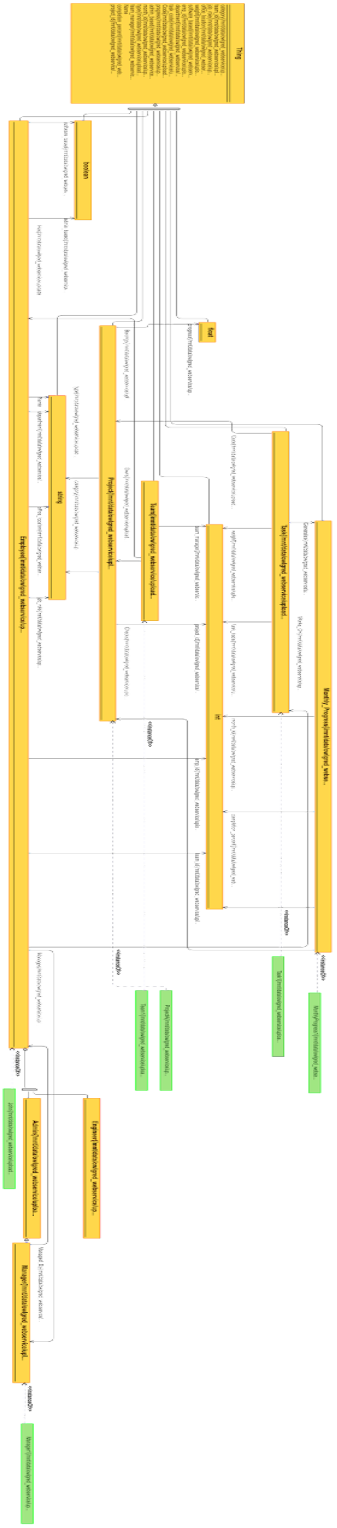
```
for $employee in /company/employees/employee
return
   <employee>
      <employee_id>{$employee/emp_id}</employee_id>
      <employee_name>{$employee/name}</employee_name>
      <office_location>{$employee/office_location}</office_location>
   </employee>
```

Result of the above expression applied to the above input file:

```
1
John Doe
London


2
Jane Smith
Glasgow


3
Michael Johnson
Manchester


4
Emily Brown
Edinburgh


5
Chris Lee
Edinburgh


```

3 Ontology

3.1 Ontology diagram with explanation



https://drive.google.com/file/d/1XlW1DMk0JuSADHh1tdoa5LWs1oQopn7M/view?usp=sharing

The Ontology that was developed was once again modelled after the SQL tables. This can be seen in the data properties; however, we have included relations and properties to add verbiage and vocab to the ontology.In terms of the flow, it follows the same hierarchy as the XML Tree.

3.2 Object properties with explanation

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <owl:Class rdf:ID="Employee"/>
   <owl:Class rdf:ID="Team"/>
   <owl:Class rdf:ID="Project"/>
   <owl:Class rdf:ID="Task"/>
   <owl:Class rdf:ID="Monthly_Progress"/>
<owl:Class rdf:about="#Admin">
         <rdfs:subClassOf rdf:resource="#Employee"/>
   </owl:Class>
   <owl:Class rdf:about="#Engineer">
         <rdfs:subClassOf rdf:resource="#Employee"/>
   </owl:Class>
   <owl:Class rdf:about="#Manager">
         <rdfs:subClassOf rdf:resource="#Admin"/>
   </owl:Class>
   <owl:ObjectProperty rdf:ID="Owns">
         <rdfs:domain rdf:resource="#Team"/>
         <rdfs:range rdf:resource="#Project"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="Works_On">
         <rdfs:domain rdf:resource="#Employee"/>
         <rdfs:range rdf:resource="#Task"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="Has">
         <rdfs:domain rdf:resource="#Team"/>
         <rdfs:range rdf:resource="#Employee"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="Checks">
         <rdfs:domain rdf:resource="#Monthly_Progress"/>
         <rdfs:range rdf:resource="#Project"/>
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="Generates">
         <rdfs:domain rdf:resource="#Task"/>
         <rdfs:range rdf:resource="#Monthly_Progress"/>
   </owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:ID="Belongs">
        <rdfs:domain rdf:resource="#Task"/>
        <rdfs:range rdf:resource="#Project"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Manages">
        <rdfs:domain rdf:resource="#Manager"/>
        <rdfs:range rdf:resource="#Employee"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="Managed_By">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="#Manager"/>
</owl:ObjectProperty>
<owl:Class rdf:about="#Employee">
        <owl:Restriction>
                <owl:onProperty rdf:resource="#Works_On"/>
                <owl:minCardinality
       rdf:datatype=
       "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
       1
       </owl:minCardinality>
        </owl:Restriction>
</owl:Class>
       <owl:Class rdf:about="#Employee">
        <owl:Restriction>
                <owl:onProperty rdf:resource="#Managed_By"/>
                <owl:maxCardinality
       rdf:datatype=
       "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
       1
       </owl:maxCardinality>
        </owl:Restriction>
</owl:Class>
<owl:Class rdf:about="#Manager">
        <owl:Restriction>
                <owl:onProperty rdf:resource="#Manages"/>
                <owl:minCardinality
       rdf:datatype=
       "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
       1
       </owl:minCardinality>
        </owl:Restriction>
</owl:Class>
<owl:Class rdf:about="#Team">
        <owl:Restriction>
                <owl:onProperty rdf:resource="#Has"/>
                <owl:minCardinality
       rdf:datatype=
       "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
       1
       </owl:minCardinality>
```

```
        </owl:Restriction>
    </owl:Class>
    <owl:Class rdf:about="#Team">
        <owl:Restriction>
            <owl:onProperty rdf:resource="#Owns"/>
            <owl:minCardinality
        rdf:datatype=
        "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
        1
        </owl:minCardinality>
        </owl:Restriction>
    </owl:Class>
    <owl:Class rdf:about="#Employee">
        <owl:Restriction>
            <owl:onProperty rdf:resource="#Works_On"/>
            <owl:minCardinality
        rdf:datatype=
        "http://www.w3.org/2001/XMLSchema#nonNegativeInteger">
        1
        </owl:minCardinality>
        </owl:Restriction>
    </owl:Class>
</rdf:RDF>
```

The Object properties server provides the vocabulary and the restrictions. The vocab and restrictions create logical connections and provide further context to the user.

3.3 Data properties with explanation

```
        <owl:DatatypeProperty rdf:about="#emp_id">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>
    <!-- Declaration of Datatype Properties -->
    <owl:DatatypeProperty rdf:about="#Name">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="#department">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="#software_based">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="&xsd;boolean"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="#admin_based">
        <rdfs:domain rdf:resource="#Employee"/>
        <rdfs:range rdf:resource="&xsd;boolean"/>
    </owl:DatatypeProperty>
```

```
<owl:DatatypeProperty rdf:about="#job_role">
	<rdfs:domain rdf:resource="#Employee"/>
	<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#team_id">
	<rdfs:domain rdf:resource="#Employee"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#office_location">
	<rdfs:domain rdf:resource="#Employee"/>
	<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#team_manager">
	<rdfs:domain rdf:resource="#Team"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#project_id">
	<rdfs:domain rdf:resource="#Project"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#type">
	<rdfs:domain rdf:resource="#Project"/>
	<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#category">
	<rdfs:domain rdf:resource="#Project"/>
	<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#progress">
	<rdfs:domain rdf:resource="#Project"/>
	<rdfs:range rdf:resource="&xsd;float"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#Code">
	<rdfs:domain rdf:resource="#Task"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#weight">
	<rdfs:domain rdf:resource="#Task"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#completion_percent">
	<rdfs:domain rdf:resource="#Monthly_Progress"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#month_id">
	<rdfs:domain rdf:resource="#Monthly_Progress"/>
	<rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#task_code">
	<rdfs:domain rdf:resource="#Monthly_Progress"/>
```

```
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>
```

The data properties serve to represent the fields in the SQL and provide key informations to the user that is making the queries.

3.4 Instances

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <!-- Individuals -->
    <owl:NamedIndividual rdf:about="#John">
        <rdf:type rdf:resource="#Employee"/>
        <emp_id rdf:datatype="http://www.w3.org/2001/XMLSchema#int">101</emp_id>
        <Name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">John
Doe</Name>
        <department
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Engineering</department>
        <software_based
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">true</software_based>
        <admin_based
rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</admin_based>
        <job_role rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Software
Engineer</job_role>
        <team_id rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</team_id>
        <office_location
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Building A, Floor
2</office_location>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="#ProjectA">
        <rdf:type rdf:resource="#Project"/>
        <project_id
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">201</project_id>
        <type rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Software
Development</type>
        <category
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Internal</category>
        <progress
rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.75</progress>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="#Task1">
        <rdf:type rdf:resource="#Task"/>
        <Code rdf:datatype="http://www.w3.org/2001/XMLSchema#int">301</Code>
```

```
        <weight rdf:datatype="http://www.w3.org/2001/XMLSchema#int">5</weight>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="#MonthlyProgress1">
        <rdf:type rdf:resource="#Monthly_Progress"/>
        <completion_percent
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">50</completion_percent>
        <month_id rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1</month_id>
        <task_code
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">301</task_code>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="#Team1">
        <rdf:type rdf:resource="#Team"/>
        <team_manager
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">102</team_manager>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="#Manager1">
        <rdf:type rdf:resource="#Manager"/>
        <Manages
rdf:datatype="http://www.w3.org/2001/XMLSchema#int">101</Manages>
    </owl:NamedIndividual>

</rdf:RDF>
```


3.5 Five SPARQL queries with results

1. List the ID and name of all employees.

```
SELECT ?projectName ?projectType
WHERE {
  ?project rdf:type ex:Project ;
        ex:Name ?projectName ;
        ex:type ?projectType .
}
```

2. List the name of all projects together with their type.

```
SELECT ?projectName ?projectType
WHERE {
  ?project rdf:type ex:Project ;
        ex:Name ?projectName ;
        ex:type ?projectType .
}
```

3. List the ID and name of all employees together with the project name and project category that
they work on.

```
SELECT ?empID ?empName ?projectName ?projectCategory
WHERE {
  ?employee rdf:type ex:Employee ;
          ex:emp_id ?empID ;
          ex:Name ?empName ;
          ex:Works_On ?project .
  ?project rdf:type ex:Project ;
         ex:Name ?projectName ;
         ex:category ?projectCategory .
}
```

4. List the ID and name of all tasks together with the name and ID of the employee who works on
this task.

```
SELECT ?taskID ?taskName ?empID ?empName
WHERE {
  ?task rdf:type ex:Task ;
       ex:Code ?taskID ;
       ex:Name ?taskName ;
       ex:Works_On ?employee .
  ?employee rdf:type ex:Employee ;
          ex:emp_id ?empID ;
          ex:Name ?empName .
}
```

5. List the ID and name of all employees together with the office number that they are based in.

```
SELECT ?empID ?empName ?officeLocation
WHERE {
  ?employee rdf:type ex:Employee ;
          ex:emp_id ?empID ;
          ex:Name ?empName ;
          ex:office_location ?officeLocation .
}
```

4 Discussion about extra knowledge based on the developed ontology

The Advantage of having the developed ontology for this scenario is that it allows us to obtain vocabulary and see relations without introducing any additional business logic. For example, while the ontology has datatype properties that are similar to the SQL and XML representation we gain vocabulary in the form of object properties. Object properties allow us to see exactly what is happening, such as “a Team *Owns* a Project” and with the addition of restrictions on the properties we can easily enforce limits, so that we know that a team will always have at least one project. We have also added inverse relationships for a manager and the whom they manage. This makes querying more efficient, as if you have the manager you can easily find who they manage and vice versa. If this was done in SQL multiple queries would be needed and additional business logic would be needed to achieve the same result.