Bachelor of Software Engineering
Centre for IT Education (CITES)
Department of Electrical and Computer Engineering
The Open University of Sri Lanka

EEI5466 – Advanced Database Systems

MINI PROJECT

09.05.2024

N. W. L. U. R. D. Nanayakkara
321424374 / s92064374
Github link: https://github.com/mendydias/5466-mini-project

https://github.com/mendydias/5466-mini-project

Table of Contents

https://github.com/mendydias/5466-mini-project

0 Overview and Environment

This document describes the overall design draft for the Employee Management System. It covers basic information like functional requirements for a few use cases, diagrams to further illustrate how the various parts of system work together, and the structure of the database

Used a docker image with MSSQL 2022 connected to JetBrains DataGrip analysis and visualisation.

Docker Compose File

```
services:
 db:
   image: mcr.microsoft.com/mssql/server:2022-latest
   restart: no
   container_name: employeedb
   volumes:
     - "./scripts:/scripts"
   ports:
     - "1433:1433"
   environment:
     - "ACCEPT_EULA=Y"
     - "MSSQL_SA_PASSWORD=Ranmal@ousl1433"
```

1 Design considerations

A database for the following scenario based on my registration number:

> **Employee Management System – Reg No (Last Digit 4 & 5)**
> A company needs to manage its employees, departments, and projects. Each employee has a unique
> ID, name, and role. Departments have a unique ID and name. Projects have a unique ID, name, and
> department ID.

This project concerns a basic employee management system. This is not production ready, this is just a proof of concept.

https://github.com/mendydias/5466-mini-project

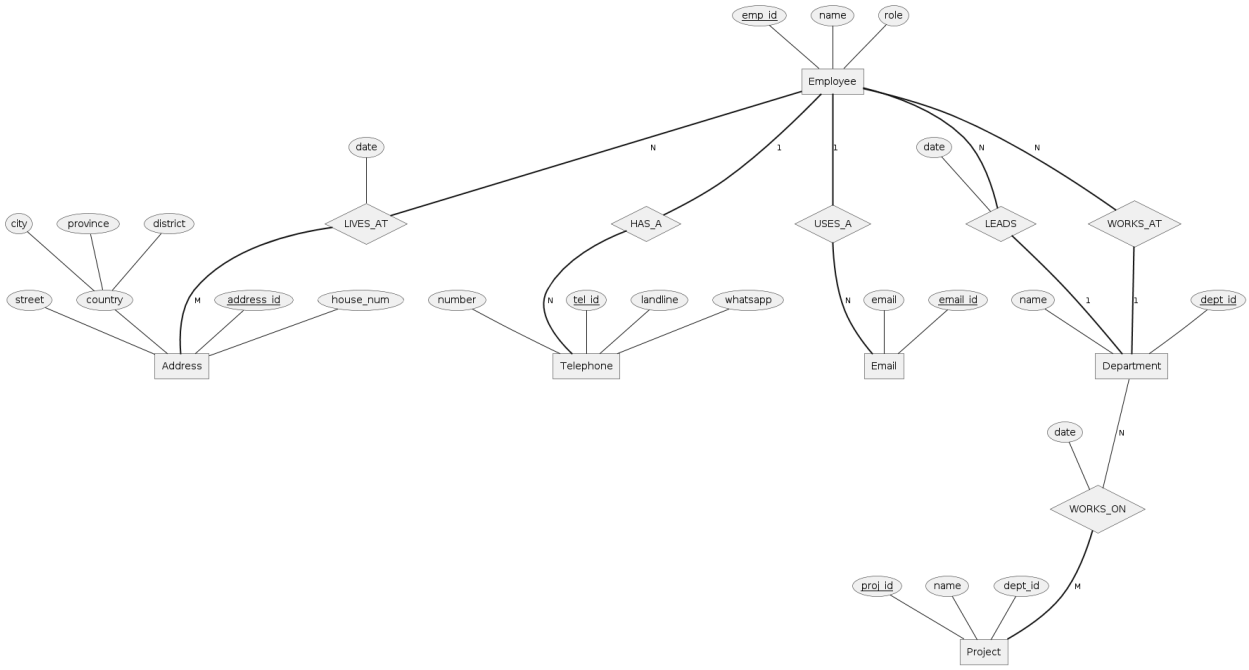
Use cases

1. As an employee, I should be able to register so that I can participate in projects.
2. As the head of a department, I should be able to assign an employee to a department so that I can delegate tasks to them.
3. As the head of the department, I should be able to assign an employee to a project so that the project can make progress.
4. As the head of the department, I should be able to see a report on the number of employees in my department so that I can make sound business decisions.
5. As the head of the department, I should be able to see a report on the projects that my department is involved in so that I can see how many of my employees are working on which projects.

Stakeholders

1. Employee
2. Project
3. Department
    a. Head of the department

Entity Relationship Diagram



https://github.com/mendydias/5466-mini-project

2 Implementation

Tables and their outputs

Employee table

	emp_id	name	role	dept_id
1	1	Saman Silva	Something	4
2	2	Chamari Perera	Developer	1
3	3	Nuwan Fernando	Designer	2
4	4	Nilmini de Silva	Analyst	3
5	5	Dinesh Gunawardena	Engineer	4
6	6	Malini Fonseka	Administrator	5
7	7	Rohan Bandara	Consultant	2
8	8	Chathuri Rajapaksa	Coordinator	6
9	9	Lakmal Perera	Specialist	6
10	10	Ishara Ranasinghe	Assistant	7
11	11	Ramesh de Alwis	Supervisor	8
12	12	Chandima Peiris	Technician	9
13	13	Lahiru Pathirana	Programmer	8
14	14	Shalini Mendis	Administrator	9
15	15	Sachith Perera	Support	7

https://github.com/mendydias/5466-mini-project

Countries table

	country_id	code	name	phone	continent	currency
1	1	AD	Andorra	376	Europe	EUR
2	2	AE	United Arab Emirates	971	Asia	AED
3	3	AF	Afghanistan	93	Asia	AFN
4	4	AG	Antigua and Barbuda	1268	North America	XCD
5	5	AI	Anguilla	1264	North America	XCD
6	6	AL	Albania	355	Europe	ALL
7	7	AM	Armenia	374	Asia	AMD
8	8	AO	Angola	244	Africa	AOA
9	9	AQ	Antarctica	672	Antarctica	noc
10	10	AR	Argentina	54	South America	ARS
11	11	AS	American Samoa	1684	Oceania	USD
12	12	AT	Austria	43	Europe	EUR
13	13	AU	Australia	61	Oceania	AUD
14	14	AW	Aruba	297	North America	AWG
15	15	AX	Aland	358	Europe	EUR
16	16	AZ	Azerbaijan	994	Asia	AZN
17	17	BA	Bosnia and Herzegovina	387	Europe	BAM
18	18	BB	Barbados	1246	North America	BBD
19	19	BD	Bangladesh	880	Asia	BDT
20	20	BE	Belgium	32	Europe	EUR
21	21	BF	Burkina Faso	226	Africa	XOF
22	22	BG	Bulgaria	359	Europe	BGN
23	23	BH	Bahrain	973	Asia	BHD
24	24	BI	Burundi	257	Africa	BIF
25	25	BJ	Benin	229	Africa	XOF
26	26	BL	Saint Barthelemy	590	North America	EUR
27	27	BM	Bermuda	1441	North America	BMD
28	28	BN	Brunei	673	Asia	BND
29	29	BO	Bolivia	591	South America	BOB
30	30	BQ	Bonaire	5997	North America	USD
31	31	BR	Brazil	55	South America	BRL
32	32	BS	Bahamas	1242	North America	BSD
33	33	BT	Bhutan	975	Asia	BTN
34	34	BV	Bouvet Island	47	Antarctica	NOK

https://github.com/mendydias/5466-mini-project

Provinces table

	province_id	name_en	name_si	name_ta	country_id
1	1	Western	????????	????	130
2	2	Central	??????	??????	130
3	3	Southern	?????	????	130
4	4	North Western	???	?? ????	130
5	5	Sabaragamuwa	???????	???????	130
6	6	Eastern	????????	???????	130
7	7	Uva	??	???	130
8	8	North Central	????? ???	?? ??????	130
9	9	Northern	?????	??	130

Districts table

	district_id	name_en	name_si	name_ta	province_id
1	1	Ampara	??????	???????	6
2	2	Anuradhapura	??????????	???????????	8
3	3	Badulla	??????	?????	7
4	4	Batticaloa	???????	????????????	6
5	5	Colombo	????	????????	1
6	6	Galle	?????	????	3
7	7	Gampaha	?????	??????	1
8	8	Hambantota	?????????	?????????????	3
9	9	Jaffna	?????	???????????	9
10	10	Kalutara	?????	?????????	1
11	11	Kandy	??????	?????	2
12	12	Kegalle	??????	??????	5
13	13	Kilinochchi	???????????	?????????	9
14	14	Kurunegala	????????	?????????	4
15	15	Mannar	???????	???????	9
16	16	Matale	?????	???????	2
17	17	Matara	????	???????	3
18	18	Monaragala	???????	????????	7
19	19	Mullaitivu	???????	????????????	9
20	20	Nuwara Eliya	???? ????	?????????	2
21	21	Polonnaruwa	??????????	?????????	8
22	22	Puttalam	???????	????????	4
23	23	Ratnapura	???????	???????????	5
24	24	Trincomalee	?????????????	?????????	6
25	25	Vavuniya	????????	???????	9

https://github.com/mendydias/5466-mini-project

Cities table

	city_id	name_en	name_si	name_ta	subname_en	subname_si	subname_ta	pos
1	1	Akkaraipattu	???????????	?????????????	<null>	<null>	<null>	32400
2	2	Ambagahawatta	????????	??????????	<null>	<null>	<null>	90326
3	3	Ampara	??????	???????	<null>	<null>	<null>	32000
4	4	Bakmitiyawa	??????????	??????????	<null>	<null>	<null>	32024
5	5	Deegawapiya	????????	???????	<null>	<null>	<null>	32006
6	6	Devalahinda	???????	???????????	<null>	<null>	<null>	32038
7	7	Digamadulla Weeragoda	?????????? ??????	?????????? ??????	<null>	<null>	<null>	32008
8	8	Dorakumbura	????????	???????????	<null>	<null>	<null>	32104
9	9	Gonagolla	????????	??????????	<null>	<null>	<null>	32064
10	10	Hulannuge	????????	???????	<null>	<null>	<null>	32514
11	11	Kalmunai	???????	???????	<null>	<null>	<null>	32300
12	12	Kannakipuram	???????????	???????????	<null>	<null>	<null>	32405
13	13	Karativu	??????	????????	<null>	<null>	<null>	32250
14	14	Kekirihena	?????????	?????????	<null>	<null>	<null>	32074
15	15	Koknahara	???????	?????????	<null>	<null>	<null>	32035
16	16	Kolamanthalawa	??????????	??????????	<null>	<null>	<null>	32102
17	17	Komari	??????	??????	<null>	<null>	<null>	32418
18	18	Lahugala	??????	??????	<null>	<null>	<null>	32512
19	19	Irakkamam	????????	?????????	<null>	<null>	<null>	32450
20	20	Mahaoya	????	??? ???	<null>	<null>	<null>	32070
21	21	Marathamune	?????????	????????	<null>	<null>	<null>	32314
22	22	Namaloya	???????	????? ??	<null>	<null>	<null>	32037
23	23	Navithanveli	???????????	???????????	<null>	<null>	<null>	32308

Addresses table

	address_id	house_num	street	city_id
1	1	123	Galle Road	349
2	2	456	Kandy Street	1678
3	3	789	Matara Lane	1203
4	4	101	Jaffna Avenue	942
5	5	202	Anuradhapura Mawatha	128
6	6	303	Polonnaruwa Drive	1319
7	7	505	Badulla Road	1567
8	8	606	Nuwara Eliya Street	1443
9	9	707	Trincomalee Avenue	1165
10	10	808	Gampaha Boulevard	187
11	11	909	Batticaloa Lane	722
12	12	111	Ratnapura Crescent	1101
13	13	222	Kurunegala Lane	769
14	14	333	Hambantota Road	305

https://github.com/mendydias/5466-mini-project

Employee Address Map

	emp_id	address_id	date
1	1	1	2023-01-01 08:45:00.000
2	2	2	2024-02-02 09:30:00.000
3	3	3	2022-03-03 10:15:00.000
4	4	4	2023-04-04 11:00:00.000
5	5	5	2024-05-05 12:15:00.000
6	6	6	2022-06-06 13:30:00.000
7	7	7	2023-07-07 14:45:00.000
8	8	8	2024-08-08 08:30:00.000
9	9	9	2022-09-09 09:45:00.000
10	10	9	2023-10-10 10:30:00.000
11	11	10	2024-11-11 11:15:00.000
12	12	11	2022-12-12 12:00:00.000
13	13	12	2023-01-13 13:15:00.000
14	14	13	2024-02-14 14:30:00.000
15	15	14	2022-03-15 08:45:00.000

Telephone table

	tel_id	number	landline	whatsapp	country_id	emp_id
1	1	1234567890	true	true	130	1
2	2	2345678901	true	false	130	2
3	3	3456789012	true	true	130	3
4	4	4567890123	true	false	130	4
5	5	5678901234	true	true	130	5
6	6	6789012345	true	false	130	6
7	7	7890123456	true	true	130	7
8	8	8901234567	true	false	130	8
9	9	9012345678	true	true	130	9
10	10	1234567890	true	false	130	10
11	11	2345678901	true	true	130	11
12	12	3456789012	true	false	130	12
13	13	4567890123	true	true	130	13
14	14	5678901234	true	false	130	14
15	15	6789012345	true	true	130	15

https://github.com/mendydias/5466-mini-project

Emails table

	email_id	email	date_added	emp_id
1	1	saman.silva@ousl.lk	2024-05-12 17:34:08.257	1
2	2	chamari.perera@ousl.lk	2024-05-12 17:34:08.257	2
3	3	nuwan.fernando@ousl.lk	2024-05-12 17:34:08.257	3
4	4	nilmini.desilva@ousl.lk	2024-05-12 17:34:08.257	4
5	5	dinesh.gunawardena@ousl.lk	2024-05-12 17:34:08.257	5
6	6	malini.fonseka@ousl.lk	2024-05-12 17:34:08.257	6
7	7	rohan.bandara@ousl.lk	2024-05-12 17:34:08.257	7
8	8	chathuri.rajapaksa@ousl.lk	2024-05-12 17:34:08.257	8
9	9	lakmal.perera@ousl.lk	2024-05-12 17:34:08.257	9
10	10	ishara.ranasinghe@ousl.lk	2024-05-12 17:34:08.257	10
11	11	ramesh.dealwis@ousl.lk	2024-05-12 17:34:08.257	11
12	12	chandima.peiris@ousl.lk	2024-05-12 17:34:08.257	12
13	13	lahiru.pathirana@ousl.lk	2024-05-12 17:34:08.257	13
14	14	shalini.mendis@ousl.lk	2024-05-12 17:34:08.257	14
15	15	sachith.perera@ousl.lk	2024-05-12 17:34:08.257	15

Departments table

	dept_id	name
1	1	Finance
2	2	Human Resources
3	3	Marketing
4	4	Sales
5	5	Research and Development
6	6	Customer Service
7	7	Information Technology
8	8	Operations
9	9	Quality Assurance

Heads of Departments map

	hod	dept_id	date_appointed	current
1	1	4	2024-04-10 10:00:00.000	true
2	2	1	2024-01-20 11:15:00.000	true
3	3	2	2022-02-23 08:45:00.000	true
4	4	3	2023-03-28 09:20:00.000	true
5	6	5	2023-05-07 16:40:00.000	true
6	9	6	2022-06-18 13:05:00.000	true
7	11	8	2023-11-15 14:30:00.000	true
8	14	7	2023-07-29 15:25:00.000	true
9	15	7	2024-07-31 08:55:00.000	true

https://github.com/mendydias/5466-mini-project

Projects

	proj_id	name
1	1	Project A
2	2	Project B
3	3	Project C
4	4	Project D
5	5	Project E
6	6	Project F
7	7	Project G
8	8	Project H
9	9	Project I
10	10	Project J
11	11	Project K
12	12	Project L
13	13	Project M
14	14	Project N
15	15	Project O

Project Assignments

	proj_id	dept_id	date_assigned
1	1	4	2029-02-18 22:20:00.000
2	2	7	2039-01-28 23:50:00.000
3	2	8	2028-10-30 20:45:00.000
4	3	9	2025-09-12 12:30:00.000
5	4	8	2023-07-28 16:40:00.000
6	5	1	2026-06-20 14:55:00.000
7	5	6	2034-03-09 01:35:00.000
8	6	9	2037-06-11 19:20:00.000
9	7	6	2031-05-25 11:50:00.000
10	8	3	2040-08-03 10:05:00.000
11	9	7	2030-11-11 09:35:00.000
12	10	5	2024-01-05 08:15:00.000
13	10	6	2038-04-16 21:35:00.000
14	11	3	2033-12-22 06:20:00.000
15	11	5	2022-03-15 10:25:00.000
16	12	2	2041-11-26 12:20:00.000
17	13	2	2027-04-02 18:10:00.000
18	13	9	2032-08-08 04:05:00.000
19	14	3	2035-07-14 03:50:00.000
20	15	1	2036-09-01 07:05:00.000

https://github.com/mendydias/5466-mini-project

Table Implementation SQL

```
create table main.employees(
   emp_id bigint identity(1,1) not null,
   name varchar(255) not null,
   role varchar(255) not null,
   constraint PK_emp_id primary key clustered (emp_id)
);
GO
--rollback drop table main.employees

--changeset mendydias:2 label:addresses
create table main.countries(
   country_id bigint not null,
   code varchar(5) not null,
   name varchar(255) not null,
   phone int,
   continent varchar(150) not null,
   currency varchar(20),
   constraint PK_country_id primary key clustered (country_id)
);
GO
--rollback drop table main.countries

create table main.provinces(
   province_id bigint not null,
   name_en varchar(255),
   name_si varchar(255),
   name_ta varchar(255),
   country_id bigint not null,
   constraint PK_province_id primary key clustered (province_id)
);
GO
--rollback drop table main.provinces

alter table main.provinces
add constraint FK_province_country
foreign key (country_id) references main.countries(country_id)
on delete cascade on update cascade;
GO
--rollback alter table main.provinces drop constraint FK_province_country

create nonclustered index IX_province_country on main.provinces (country_id);
GO
--rollback drop index IX_province_country on main.provinces

create table main.districts(
   district_id bigint not null,
   name_en varchar(255),
```

https://github.com/mendydias/5466-mini-project

```
   name_si varchar(255),
   name_ta varchar(255),
   province_id bigint not null,
   constraint PK_district_id primary key clustered (district_id)
);
GO
--rollback drop table main.districts

alter table main.districts
add constraint FK_district_province
foreign key (province_id) references main.provinces(province_id)
on delete cascade on update cascade;
GO
--rollback alter table main.districts drop constraint FK_district_province

create nonclustered index IX_district_province on main.districts(province_id);
GO
--rollback drop index IX_district_province on main.districts

create table main.cities(
   city_id bigint not null,
   name_en varchar(255),
   name_si varchar(255),
   name_ta varchar(255),
   subname_en varchar(255),
   subname_si varchar(255),
   subname_ta varchar(255),
   postcode varchar(50),
   latitude varchar(150),
   longitude varchar(150),
   district_id bigint not null,
   constraint PK_city_district primary key clustered (city_id)
);
GO
--rollback drop table main.cities

alter table main.cities
add constraint FK_city_district
foreign key (district_id) references main.districts(district_id)
on delete cascade on update cascade;
GO
--rollback alter table main.cities drop constratin FK_city_district

create nonclustered index IX_city_district on main.cities (district_id);
GO
--rollback drop index IX_city_district on main.cities

create table main.addresses(
   address_id bigint identity(1,1) not null,
```

https://github.com/mendydias/5466-mini-project

```
   house_num varchar(50) not null,
   street varchar(255) not null,
   city_id bigint not null,
   constraint PK_address_id primary key clustered (address_id)
);
GO
--rollback drop table main.addresses

alter table main.addresses
add constraint FK_city_address
foreign key (city_id) references main.cities(city_id)
on delete cascade on update cascade;
GO
--rollback alter table main.addresses drop constraint FK_city_address

create nonclustered index IX_address_city on main.addresses (city_id);
GO
--rollback drop index IX_address_city on main.addresses

--changeset mendydias:3 label:connect_addresses_to_employees
create table main.emp_addresses(
   emp_id bigint not null,
   address_id bigint not null,
   date datetime default getdate(),
   constraint PK_emp_address primary key clustered (emp_id, address_id)
);
GO
--rollback drop table emp_addresses

alter table main.emp_addresses
add constraint FK_employee_addresses
foreign key (emp_id) references main.employees (emp_id)
on delete cascade on update cascade;
GO
--rollback alter table main.emp_addresses drop constraint
FK_employee_addresses

alter table main.emp_addresses
add constraint FK_addresses_employee
foreign key (address_id) references main.addresses (address_id)
on delete cascade on update cascade;
GO
--rollback alter table main.emp_addresses drop constraint
FK_addresses_employee

--changeset mendydias:4 label:telephone
create table main.telephone(
   tel_id bigint identity(1,1) not null,
   number bigint,
```

https://github.com/mendydias/5466-mini-project

```
   landline bit default 0,
   whatsapp bit default 0,
   country_id bigint not null,
   emp_id bigint not null,
   constraint PK_telephone primary key clustered (tel_id)
);
GO
--rollback drop table main.telephone

alter table main.telephone
add constraint FK_tel_country
foreign key (country_id) references main.countries (country_id)
on delete cascade on update cascade;
GO
--rollback alter table main.telephone drop constraint FK_tel_country

alter table main.telephone
add constraint FK_tel_employee
foreign key (emp_id) references main.employees (emp_id)
on delete cascade on update cascade;
GO
--rollback alter table main.telephone drop constraint FK_tel_employee

create nonclustered index IX_tel_country on main.telephone (country_id);
GO
--rollback drop index IX_tel_country on main.telephone

create nonclustered index IX_tel_employee on main.telephone (emp_id);
GO
--rollback drop index IX_tel_employee on main.telephone

--changeset mendydias:5 label:email
create table main.emails(
   email_id bigint identity(1,1) not null,
   email varchar(150) not null,
   date_added datetime not null default getdate(),
   emp_id bigint not null,
   constraint PK_emails primary key clustered (email_id)
);
GO
--rollback drop table main.emails

alter table main.emails
add constraint FK_emails_employee
foreign key (emp_id) references main.employees (emp_id)
on delete cascade on update cascade;
GO
--rollback alter table main.emails drop constraint FK_emails_employee
```

https://github.com/mendydias/5466-mini-project

```
create nonclustered index IX_emails_emp on main.emails (emp_id);
GO
--rollback drop index IX_emails_emp on main.emails

--changeset mendydias:6 label:departments
create table main.departments(
   dept_id bigint identity(1,1) not null,
   name varchar(255) not null,
   constraint PK_departments primary key clustered (dept_id)
);
GO
--rollback drop table main.departments

create table main.dept_hods(
   hod bigint not null,
   dept_id bigint not null,
   date_appointed datetime not null,
   constraint PK_dept_hod primary key clustered (hod, dept_id)
);
GO
--rollback drop table main.dept_hods

alter table main.dept_hods
add constraint FK_dept_employee
foreign key (hod) references main.employees (emp_id)
on delete cascade on update cascade;
GO
--rollback alter table main.dept_hods drop constraint FK_dept_employee

alter table main.dept_hods
add constraint FK_hod_dept
foreign key (dept_id) references main.departments (dept_id)
on delete cascade on update cascade;
GO
--rollback alter table main.dept_hods drop constraint FK_hod_dept

alter table main.employees add dept_id bigint;
GO
--rollback alter table main.employee drop column dept_id
alter table main.employees
add constraint FK_emp_dept
foreign key (dept_id) references main.departments (dept_id)
GO
--rollback alter table main.employees drop constraint FK_emp_dept

create nonclustered index IX_emp_dept on main.employees (dept_id);
--rollback drop index IX_emp_dept on main.employees

--changeset mendydias:7 label:project
```

https://github.com/mendydias/5466-mini-project

```sql
create table main.projects(
   proj_id bigint identity(1,1) not null,
   name varchar(255) not null,
   constraint PK_projects primary key clustered (proj_id)
);
GO
--rollback drop table main.projects

create table main.project_assignments(
   proj_id bigint not null,
   dept_id bigint not null,
   date_assigned datetime not null default getdate(),
   constraint PK_proj_assign primary key clustered (proj_id, dept_id)
);
GO
--rollback drop table main.project_assignments

alter table main.project_assignments
add constraint FK_projects
foreign key (proj_id) references main.projects (proj_id)
on delete cascade on update cascade;
GO
--rollback alter table drop constraint FK_projects

alter table main.project_assignments
add constraint FK_departments
foreign key (dept_id) references main.departments (dept_id)
on delete cascade on update cascade;
GO
--rollback alter table drop constraint FK_departments
```

https://github.com/mendydias/5466-mini-project

3 Common operations

Some common operations will be executed on the database set up above as transactions. Each operation will be started with the BEGIN TRANSACTION statement. This marks the point at which the data referenced by the transaction is logically and physically consistent. If anything goes wrong during the transaction operations, then the state of the data is reverted back to the state at the BEGIN TRANSACTION statement. The changes made during the transaction, however, are not made permanent until the database reaches a commit statement successfully. If any errors occur during the transaction, the changes are rolled back.

For the business use case, the following common operations will be handled:

1. **Registering an employee and adding to a department, then promoting the employee as the head of a department**.

This business function includes the following operations: an insert, an update and a retrieval.

```
BEGIN TRANSACTION;

INSERT INTO main.employees(name, role, dept_id)
VALUES ('Ranmal Dias', 'Mediator', 4);

UPDATE main.dept_hods
SET current_hod = 0
WHERE hod = (SELECT TOP(1) hod FROM main.dept_hods WHERE dept_id
= 4 ORDER BY date_appointed DESC);

INSERT INTO main.dept_hods(hod, dept_id, date_appointed,
current_hod)
VALUES (16, 4, GETDATE(), 1);

SELECT * FROM main.dept_hods;

COMMIT;
```

https://github.com/mendydias/5466-mini-project

	emp_id	name	role	dept_id
1	1	Saman Silva	Something	4
2	2	Chamari Perera	Developer	1
3	3	Nuwan Fernando	Designer	2
4	4	Nilmini de Silva	Analyst	3
5	5	Dinesh Gunawardena	Engineer	4
6	6	Malini Fonseka	Administrator	5
7	7	Rohan Bandara	Consultant	2
8	8	Chathuri Rajapaksa	Coordinator	6
9	9	Lakmal Perera	Specialist	6
10	10	Ishara Ranasinghe	Assistant	7
11	11	Ramesh de Alwis	Supervisor	8
12	12	Chandima Peiris	Technician	9
13	13	Lahiru Pathirana	Programmer	8
14	14	Shalini Mendis	Administrator	9
15	15	Sachith Perera	Support	7
16	16	Ranmal Dias	Mediator	4

	hod	dept_id	date_appointed	current_hod
1	1	4	2024-04-10 10:00:00.000	false
2	2	1	2024-01-20 11:15:00.000	• true
3	3	2	2022-02-23 08:45:00.000	• true
4	4	3	2023-03-28 09:20:00.000	• true
5	6	5	2023-05-07 16:40:00.000	• true
6	9	6	2022-06-18 13:05:00.000	• true
7	11	8	2023-11-15 14:30:00.000	• true
8	14	7	2023-07-29 15:25:00.000	• true
9	15	9	2024-07-31 08:55:00.000	• true
10	16	4	2024-05-13 07:14:55.707	• true

The beauty of handling transactions this way is shown just from this sample operation; I made a syntax error in the update, however multiple inserts were not made in the process of executing this sql transaction as the errors rolled back the previous statements regardless of whether they were correct.

https://github.com/mendydias/5466-mini-project

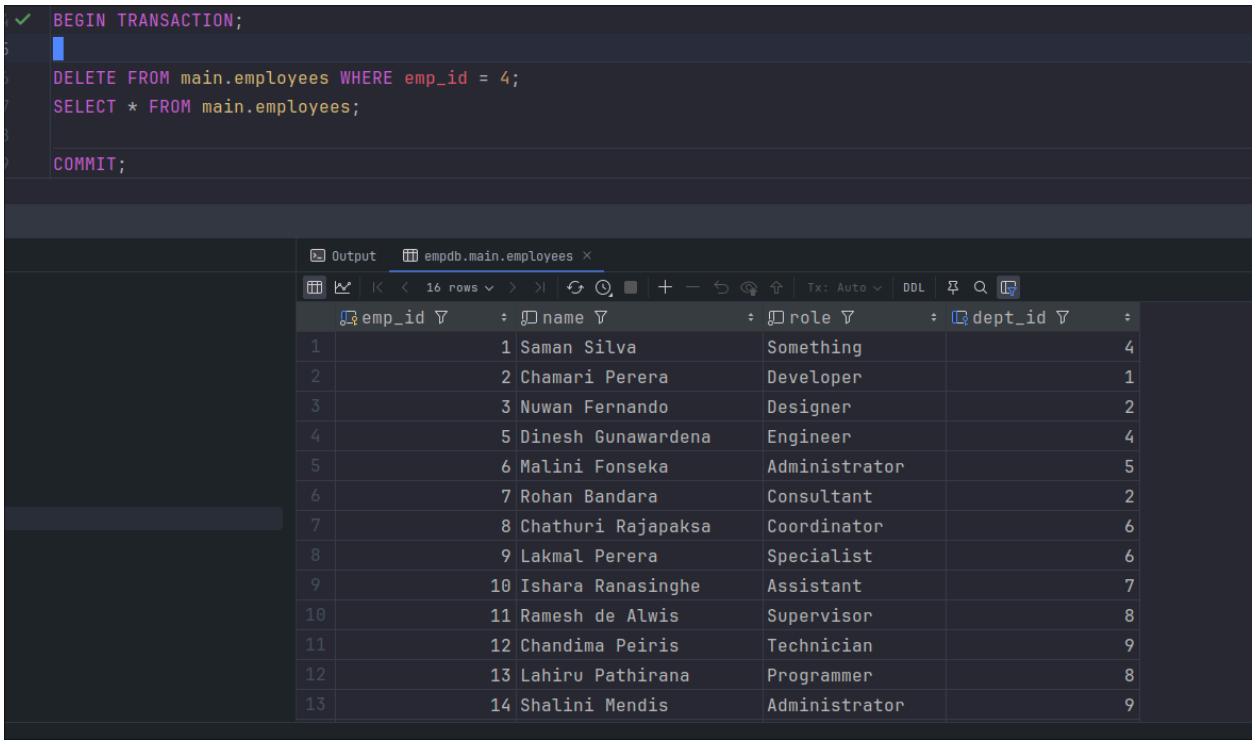
2. **Deleting an employee and their projects**

This business function includes the following operations: a delete and a retrieval (to show the results of the update operation)

```
BEGIN TRANSACTION;

DELETE FROM main.employees WHERE emp_id = 4;
SELECT * FROM main.employees;

COMMIT;
```



	emp_id	name	role	dept_id
1	1	Saman Silva	Something	4
2	2	Chamari Perera	Developer	1
3	3	Nuwan Fernando	Designer	2
4	5	Dinesh Gunawardena	Engineer	4
5	6	Malini Fonseka	Administrator	5
6	7	Rohan Bandara	Consultant	2
7	8	Chathuri Rajapaksa	Coordinator	6
8	9	Lakmal Perera	Specialist	6
9	10	Ishara Ranasinghe	Assistant	7
10	11	Ramesh de Alwis	Supervisor	8
11	12	Chandima Peiris	Technician	9
12	13	Lahiru Pathirana	Programmer	8
13	14	Shalini Mendis	Administrator	9

4 Advanced Business Functions

The following scenarios will be encapsulated in a stored procedure and a function.

1. **Hiring a new employee (possible HR business function)**

The process goes something like this:

- Insert a new employee record into the employees table with provided details.
- Insert a new address record into the addresses table with the address details provided.
- Insert a new record into the emp_addresses table linking the employee and address.

https://github.com/mendydias/5466-mini-project

- Insert a new telephone record into the telephone table with country and employee linkage.
- Insert a new email record into the emails table linking it to the employee.

https://github.com/mendydias/5466-mini-project

Implementation

```
CREATE PROCEDURE main.register(@name varchar(255), @role
varchar(255),
@department bigint, @house_num varchar(50), @street
varchar(255), @city_id bigint,
@email varchar(150), @telephone bigint)
AS
BEGIN
   SET NOCOUNT ON
   BEGIN TRANSACTION;

   -- save employee details
   INSERT INTO main.employees (name, role, dept_id) VALUES
(@name, @role, @department);

   DECLARE @new_id bigint;
   SELECT @new_id = emp_id
   FROM main.employees
   WHERE name = @name;

   -- save address
   INSERT INTO main.addresses(house_num, street, city_id) VALUES
(@house_num, @street, @city_id);
   DECLARE @new_address_id bigint;
   SELECT @new_address_id = address_id FROM main.addresses WHERE
house_num = @house_num;

   -- link the two addresses
   INSERT INTO main.emp_addresses(emp_id, address_id) VALUES
(@new_id, @new_address_id);

   -- save telephone number
   DECLARE @country_id bigint;
   SELECT @country_id = p.country_id
   FROM main.cities ci
            INNER JOIN main.districts d ON ci.district_id =
d.district_id
            INNER JOIN main.provinces p on p.province_id =
d.province_id
   WHERE ci.city_id = @city_id;
```

https://github.com/mendydias/5466-mini-project

```
   INSERT INTO main.telephone(number, country_id, emp_id) VALUES
(@telephone, @country_id, @new_id);

   -- save email address
   INSERT INTO main.emails (email, date_added, emp_id) VALUES
(@email, GETDATE(), @new_id);
   COMMIT;

END;
GO
```

Execution

```
EXECUTE main.register N'Harriet Tubman', N'Cook', 3, N'223/12',
N'Boralesgamuwa avenue', 280, N'h.tubman@ousl.lk', 662800446;
GO
```

Results

	emp_id	name	role	dept_id
1	1	Saman Silva	Something	4
2	2	Chamari Perera	Developer	1
3	3	Nuwan Fernando	Designer	2
4	5	Dinesh Gunawardena	Engineer	4
5	6	Malini Fonseka	Administrator	5
6	7	Rohan Bandara	Consultant	2
7	8	Chathuri Rajapaksa	Coordinator	6
8	9	Lakmal Perera	Specialist	6
9	10	Ishara Ranasinghe	Assistant	7
10	11	Ramesh de Alwis	Supervisor	8
11	12	Chandima Peiris	Technician	9
12	13	Lahiru Pathirana	Programmer	8
13	14	Shalini Mendis	Administrator	9
14	15	Sachith Perera	Support	7
15	16	Ranmal Dias	Mediator	4
16	17	Ranmal Dias	Mediator	4
17	18	Harriet Tubman	Cook	3

https://github.com/mendydias/5466-mini-project

	address_id	house_num	street	city_id
1	1	123	Galle Road	349
2	2	456	Kandy Street	1678
3	3	789	Matara Lane	1203
4	4	101	Jaffna Avenue	942
5	5	202	Anuradhapura Mawatha	128
6	6	303	Polonnaruwa Drive	1319
7	7	505	Badulla Road	1567
8	8	606	Nuwara Eliya Street	1443
9	9	707	Trincomalee Avenue	1165
10	10	808	Gampaha Boulevard	187
11	11	909	Batticaloa Lane	722
12	12	111	Ratnapura Crescent	1101
13	13	222	Kurunegala Lane	769
14	14	333	Hambantota Road	305
15	15	223/12	Boralesgamuwa avenue	280

	emp_id	address_id	date
1	1	1	2023-01-01 08:45:00.000
2	2	2	2024-02-02 09:30:00.000
3	3	3	2022-03-03 10:15:00.000
4	5	5	2024-05-05 12:15:00.000
5	6	6	2022-06-06 13:30:00.000
6	7	7	2023-07-07 14:45:00.000
7	8	8	2024-08-08 08:30:00.000
8	9	9	2022-09-09 09:45:00.000
9	10	9	2023-10-10 10:30:00.000
10	11	10	2024-11-11 11:15:00.000
11	12	11	2022-12-12 12:00:00.000
12	13	12	2023-01-13 13:15:00.000
13	14	13	2024-02-14 14:30:00.000
14	15	14	2022-03-15 08:45:00.000
15	18	15	2024-05-13 08:46:16.867

https://github.com/mendydias/5466-mini-project

	email_id	email	date_added	emp_id
1	1	saman.silva@ousl.lk	2024-05-13 07:13:18.170	1
2	2	chamari.perera@ousl.lk	2024-05-13 07:13:18.170	2
3	3	nuwan.fernando@ousl.lk	2024-05-13 07:13:18.170	3
4	5	dinesh.gunawardena@ousl.lk	2024-05-13 07:13:18.170	5
5	6	malini.fonseka@ousl.lk	2024-05-13 07:13:18.170	6
6	7	rohan.bandara@ousl.lk	2024-05-13 07:13:18.170	7
7	8	chathuri.rajapaksa@ousl.lk	2024-05-13 07:13:18.170	8
8	9	lakmal.perera@ousl.lk	2024-05-13 07:13:18.170	9
9	10	ishara.ranasinghe@ousl.lk	2024-05-13 07:13:18.170	10
10	11	ramesh.dealwis@ousl.lk	2024-05-13 07:13:18.170	11
11	12	chandima.peiris@ousl.lk	2024-05-13 07:13:18.170	12
12	13	lahiru.pathirana@ousl.lk	2024-05-13 07:13:18.170	13
13	14	shalini.mendis@ousl.lk	2024-05-13 07:13:18.170	14
14	15	sachith.perera@ousl.lk	2024-05-13 07:13:18.170	15
15	16	h.tubman@ousl.lk	2024-05-13 08:46:16.867	18

	tel_id	number	landline	whatsapp	country_id	emp_id
1	1	1234567890	true	true	130	1
2	2	2345678901	true	false	130	2
3	3	3456789012	true	true	130	3
4	5	5678901234	true	true	130	5
5	6	6789012345	true	false	130	6
6	7	7890123456	true	true	130	7
7	8	8901234567	true	false	130	8
8	9	9012345678	true	true	130	9
9	10	1234567890	true	false	130	10
10	11	2345678901	true	true	130	11
11	12	3456789012	true	false	130	12
12	13	4567890123	true	true	130	13
13	14	5678901234	true	false	130	14
14	15	6789012345	true	true	130	15
15	16	662800446	false	false	130	18

https://github.com/mendydias/5466-mini-project

2. **Find out how long an employee has been working at the Department(e.g. through a TENURE function)**

The process is as follows:

- Access the department head table and retrieve the date_added field for the specified employee ID.
- Calculate the difference between the retrieved date_added and the current date.
- The function should return the calculated tenure.

Implementation

```
-- Calculate working period
CREATE FUNCTION main.TENURE(@emp_id bigint)
   RETURNS INT
   WITH EXECUTE AS CALLER
AS
BEGIN
   DECLARE @date DATETIME;
   SELECT @date = date_appointed FROM dept_hods WHERE hod =
@emp_id;
   RETURN DATEDIFF(month, @date, GETDATE());
END
GO
```

Execution

```
SELECT main.TENURE(6) as "Number of months as Head";
```

Results

	[Number of months as Head]
1	12

https://github.com/mendydias/5466-mini-project

5 Java CRUD

The program is implemented as a simple cli. It will prompt the user for a command and then call the respective stored procedure or function through the DB service.

Implementation

App.java

```
package org.example;

public class App {
   public static void main(String[] args) {
       Composer composer = new Composer();

       composer.greet();
       composer.play();
   }
}
```

Employee

```
package org.example;

public record EmployeeDTO(String name, String role, Long dept,
String houseNum, String street, Long cityId,
       String email, Long tel) {
}
```

https://github.com/mendydias/5466-mini-project

DbService.java

```
package org.example;

import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class DbService {
   private static final String URL =
"jdbc:sqlserver://localhost;database=empdb;encrypt=true;trustSer
verCertificate=true";
   private static final String USER = "sa";
   private static final String PASS = "Ranmal@ousl1433";

   public void callRegisterProc(EmployeeDTO employeeDTO) {
       try (Connection conn = DriverManager.getConnection(URL,
USER, PASS)) {
           String procedure = "{call
main.register(?,?,?,?,?,?,?,?)}";
           CallableStatement stmt = conn.prepareCall(procedure);
           stmt.setString("name", employeeDTO.name());
           stmt.setString("role", employeeDTO.role());
           stmt.setLong("department", employeeDTO.dept());
           stmt.setString("house_num", employeeDTO.houseNum());
           stmt.setString("street", employeeDTO.street());
           stmt.setLong("city_id", employeeDTO.cityId());
           stmt.setString("email", employeeDTO.email());
           stmt.setLong("telephone", employeeDTO.tel());
           stmt.execute();
       } catch (SQLException sqe) {
           sqe.printStackTrace();
       }
   }

   public String calculateTenure(Long empId) {
       try (Connection conn = DriverManager.getConnection(URL,
USER, PASS)) {
           String result = "";
```

https://github.com/mendydias/5466-mini-project

```
            String tenureFunc = "SELECT main.TENURE(?) as
tenure;";
            try (PreparedStatement stmt =
conn.prepareStatement(tenureFunc)) {
                stmt.setLong(1, empId);
                try (ResultSet results = stmt.executeQuery()) {
                    while (results.next()) {
                        result += results.getInt("tenure");
                    }
                }
            }
            return result.equals("1") ? result + " month" :
result + " months";
        } catch (SQLException sqe) {
            sqe.printStackTrace();
            return "-1";
        }
    }
}
```

Composer.java

```
package org.example;

import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

class AppState {
    private boolean running = true;

    public boolean isRunning() {
        return running;
    }

    public void stop() {
        running = false;
    }
}

public class Composer {
```

https://github.com/mendydias/5466-mini-project

```
    private static final Map<String, Command> CMD = new
HashMap<>();
    private final AppState state = new AppState();

    public Composer() {
        CMD.put("register", new RegisterCommand(new
DbService()));
        CMD.put("calcTenure", new CalculateTenureCommand(new
DbService()));
        CMD.put("quit", new QuitCommand(() -> state.stop()));
        CMD.put("help", new HelpCommand(CMD));
    }

    public void greet() {
        System.out.println("Simple Employee Management System");
        System.out.println("Name: N. W. L. U. R. D.
Nanayakakra");
        System.out.println("Reg. no.: 321424374");
        System.out.println("\nType your commands after the
prompt. Type help to see available commands.");
    }

    public void play() {
        try (Scanner input = new Scanner(System.in)) {
            while (state.isRunning()) {
                System.out.print("> ");
                parseExecute(input.next());
            }
        }
    }

    private void parseExecute(String cmd) {
        Command command = CMD.getOrDefault(cmd, new
DefaultCommand());

        System.out.println(command.execute() + "\n");
    }
}
```

Command Interface and Various Commands

```
package org.example;
```

https://github.com/mendydias/5466-mini-project

```
public interface Command {
   String execute();
   String desc();
}
```

https://github.com/mendydias/5466-mini-project

```
package org.example;

import java.util.Scanner;

/**
* RegisterCommand
*/
public class RegisterCommand implements Command {
   private EmployeeDTO employee;

   private final DbService service;

   public RegisterCommand(DbService service) {
       this.service = service;
   }

   private void init() {
       Scanner input = new Scanner(System.in);
       System.out.println("Enter the following information to
register:");
       System.out.println("Full Name:");
       String name = input.nextLine();
       System.out.println("Role:");
       String role = input.nextLine();
       System.out.println("Department id:");
       Long dept = Long.parseLong(input.nextLine());
       System.out.println("House number:");
       String houseNum = input.nextLine();
       System.out.println("Street:");
       String street = input.nextLine();
       System.out.println("City id:");
       Long cityId = Long.parseLong(input.nextLine());
       System.out.println("Email:");
       String email = input.nextLine();
       System.out.println("Telephone number:");
       Long tel = Long.parseLong(input.nextLine());

       employee = new EmployeeDTO(name, role, dept, houseNum,
street, cityId, email, tel);
   }
```

https://github.com/mendydias/5466-mini-project

```
    @Override
    public String execute() {
        init();
        service.callRegisterProc(employee);
        return "";
    }

    @Override
    public String desc() {
        return "- this command registers a new user in the
Employee Management Database";
    }
}
```

```
package org.example;

import java.util.Scanner;

public class CalculateTenureCommand implements Command {
    private final DbService service;

    public CalculateTenureCommand(DbService service) {
        this.service = service;
    }

    @Override
    public String execute() {
        System.out.println("Enter Employee id:");
        Scanner input = new Scanner(System.in);
        Long empId = Long.parseLong(input.nextLine());
        return service.calculateTenure(empId);
    }

    @Override
    public String desc() {
        return "- This calculates the number of months the
employee has worked as the head of the department.";
    }
}
```

```
package org.example;
```

https://github.com/mendydias/5466-mini-project

```
public class DefaultCommand implements Command {
   @Override
   public String execute() {
       return "Unexpected command. Type help to see a list of
available commands or quit to exit.\n";
   }

   @Override
   public String desc() {
       return null;
   }
}
```

```
package org.example;

public class QuitCommand implements Command {
   private final Runnable callback;

   public QuitCommand(Runnable callback) {
       this.callback = callback;
   }

   @Override
   public String execute() {
       callback.run();
       return "";
   }

   @Override
   public String desc() {
       return "- Quits the program.";
   }
}
```

```
package org.example;

import java.util.Map;

public class HelpCommand implements Command {
   private final Map<String, Command> commands;
```

https://github.com/mendydias/5466-mini-project

```
    public HelpCommand(Map<String, Command> commands) {
        this.commands = commands;
    }

    @Override
    public String execute() {
        StringBuilder helpmsg = new StringBuilder();
        helpmsg.append("The following commands are
available:\n\n");
        for (Map.Entry<String, Command> command :
commands.entrySet()) {
            String description = command.getValue().desc();
            if (description != null) {
                helpmsg.append(command.getKey() + " " +
description + "\n");
            }
        }
        return helpmsg.append("\n")
                .toString();
    }

    @Override
    public String desc() {
        return "- Print out this message.";
    }
}
```

https://github.com/mendydias/5466-mini-project

Results