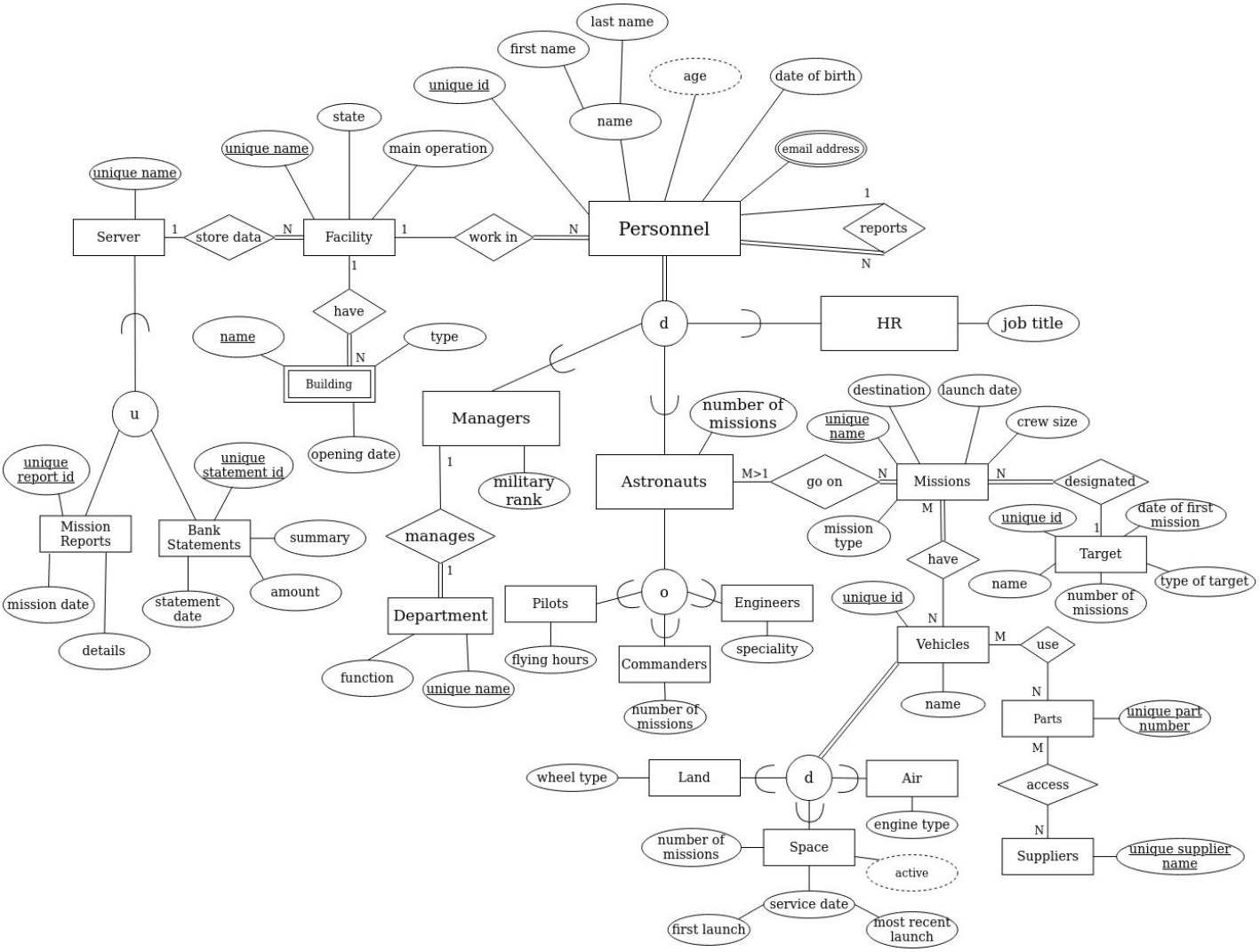
Coursework 1 - Database Systems

Part 1:

a. EER of the NASA database



Report:

The starting point of this EER was the entity called Personnel, as every other entity was in one way or another connected to it. Because each Personnel must report to another Personnel, and a Personnel can have multiple Personnel report to them, I have made a returning relation between it. From there, there are two main important sequences it goes to, the Facility where the Personnel work in, and the type of each Personnel, of which there can be only one.

The Facility route is pretty straight forward, as it can have Buildings, and must store its data on a Server, which contains two data types, Mission Reports and Bank Statements.

Each Personnel can be put in one of each 3 types, Manager, Astronaut or HR. HR's are simple and don't have any more connections. Managers can manage one more Departments.

Furthermore, Astronauts can be one or more of 3 types, Pilots, Commanders or Engineers. Each one of those Astronauts can go on Missions, which must have a Target. Next, each Mission must have multiple Vehicles, which can be one of 3 types, Land, Air or Space. A Vehicle can use multiple Parts, accessed by multiple Suppliers.

As for the design, I have chosen to start from Personnel and then expand into each type separately, after being done with each type tree and following every instruction and detail, Managers, Astronauts and HR had, with Astronauts being the longest, as it had more types and Missions with Targets and Vehicles. The other part was creating the Facility route, which was mostly separate from the other types disjoint with Personnel. The Facility was fairly short as it had only 2 entities going from it, and 2 more from the server.

In the end, I went over the flowchart again to make sure it's readable and rearranged it in a more organized way, so that it is easier to go through.

Part 1:

B. Relational Schema

Personnel (personnel_ID, facilityName, name, dateOfBirth, age, first_name, last_name, email_address):
    pk[personnel_ID]
    fk[facilityName -> Facility.facilityName]

Managers (manager_ID, militaryRank):
    pk[manager_ID]
    fk[manager_ID -> Personnel.personnel_ID]

HR (HR_ID, jobTitle):
    pk[HR_ID]
    fk[HR_ID -> Personnel.personnel_ID]

Astronauts (astronauts_ID, numberOfMissions):
	pk[astronauts_ID]
	fk[astronauts_ID -> Personnel.personnel_ID]

Department (department_name, manager_ID, function):
	pk[department_name]
	fk[manager_ID -> Managers.manager_ID]

Pilots (pilot_ID, flyingHours):
	pk[pilot_ID]
	fk[pilot_ID -> Astronauts.astronauts_ID]

Commanders (commander_ID, numberOfMissions):
	pk[commander_ID]
	fk[commander_ID -> Astronauts.astronauts_ID]

Engineers (engineer_ID, speciality):
	pk[engineer_ID]
	fk[engineer_ID -> Astronauts.astronauts_ID]

Missions (missionName, astronauts_ID, destination, launchDate, crewSize):
	pk[missionName]
	fk[astronauts_ID -> Astronauts.astronauts_ID]

Target (target_ID, missionName, name, dateOfFirstMission, typeOfTarget, numberOfMissions):
	pk[target_ID]
	fk[missionName -> Missions.missionName]

Vehicles (vehicles_ID, missionName, name):
	pk[vehicles_ID]
	fk[missionName -> Mission.missionName]

Land (landV_ID, wheelType):
	pk[landV_ID]
	fk[landV_ID -> Vehicles.vehicles_ID]

Space (spaceV_ID, serviceDate, firstLaunch, numberOfMissions, mostRecentLaunch, active):
	pk[spaceV_ID]
	fk[spaceV_ID -> Vehicles.vehicles_ID]

Air (airV_ID, engineType):
	pk[airV_ID]
	fk[airV_ID -> Vehicles.vehicles_ID]

Parts (partNumber, vehicles_ID):
	pk[partNumber]
	fk[vehicles_ID -> Vehicles.vehicles_ID]

Suppliers (supplierName, partNumber):
	pk[supplierName]
	fk[partNumber -> Parts.partNumber]

Facility (facilityName, state, mainOperation):
pk[facilityName]

Building (buildingName, facilityName, type, openingDate):
pk[buildingName]
fk[facilityName -> Facility.facilityName]

Server (serverName):
pk[serverName]
fk[facilityName -> Facility.facilityName]

Mission_Reports (report_ID, serverName, missionDate, details):
pk[report_ID]
fk[serverName -> Server.serverName]

Bank_Statements (statement_ID, serverName, statementDate, amount, summary):
pk[statement_ID]
fk[serverName -> Server.serverName]

Report:

When creating the relational schema, I made sure I noted every primary key and foreign key first, so that I know where to go from. Then I looked over the example we were given in the lectures and thought I should make something similar to it.

I had to make sure that each table has a primary key, so that I can connect it to its parent table, with a foreign key, except for the Facility table, for which I could not think of any foreign key.

Each foreign key -> primary key connection is made exactly based on the EER we drew in the step before this one, with the table name right behind the primary key in the connection, so that it is easy to follow where each primary or foreign key goes.

The attributes of all the tables are included in the parentheses next to the table names.

For those tables who did not have a primary key, but I thought they would need one, as to make a connection with other tables, I created an ID (primary key) for them.

I have tried my best to write and put the schema in a way that would be easy to go through and understand, with each connection being simple and straight-forward.

Part 2:

a. Documentation

Student		
Field	**Data Type**	**Constraints**
id	Integer	Unsigned, 0-4294967295, primary key, not null
name	VarChar	255 characters, not null
dept_name	VarChar	100 characters, foreign key, not null
tot_cred	SmallInt	Unsigned, 0-65535

Takes		
Field	**Data Type**	**Constraints**
id	Integer	Unsigned, 0-4294967295, foreign key, not null
course_id	VarChar	255 characters, foreign key, not null
sec_id	Integer	Unsigned, 0-4294967295, foreign key, not null
semester	VarChar	100 characters, foreign key, not null
year	Integer	Unsigned, 0-4294967295, foreign key, not null
grade	VarChar	100 characters

Section		
Field	**Data Type**	**Constraints**
course_id	VarChar	255 characters, primary and foreign key, not null
sec_id	Integer	Unsigned, 0-4294967295, primary key, not null
semester	VarChar	100 characters, primary key, not null
year	Integer	Unsigned, 0-4294967295, primary key, not null
buildings	VarChar	255 characters, foreign key
room_no	Integer	Unsigned, 0-4294967295, foreign key
time_slot_id	VarChar	255 characters, foreign key

Classroom		
Field	**Data Type**	**Constraints**
buildings	VarChar	255 characters, primary key, not null
room_no	Integer	Unsigned, 0-4294967295, primary key, not null
capacity	Integer	Unsigned, 0-4294967295

Time_slot		
Field	**Data Type**	**Constraints**
time_slot_id	VarChar	255 characters, primary key, not null
day	VarChar	100 characters, not null
start_hour	Integer	Unsigned, 0-4294967295, not null
start_min	Integer	Unsigned, 0-4294967295, not null
end_hour	Integer	Unsigned, 0-4294967295
end_min	Integer	Unsigned, 0-4294967295

Teaches		
Field	**Data Type**	**Constraints**
id	Integer	Unsigned, 0-4294967295, foreign key, not null
course_id	VarChar	255 characters, foreign key, not null
sec_id	Integer	Unsigned, 0-4294967295, foreign key, not null
semester	VarChar	100 characters, foreign key, not null
year	Integer	Unsigned, 0-4294967295, foreign key, not null

Course		
Field	**Data Type**	**Constraints**
course_id	VarChar	255 characters, primary key, not null
title	VarChar	255 characters
dept_name	VarChar	100 characters, foreign key
credits	Integer	Unsigned, 0-4294967295

Prereq		
Field	**Data Type**	**Constraints**
course_id	VarChar	100 characters, foreign key, not null
prereq_id	VarChar	100 characters, foreign key, not null

Department		
Field	**Data Type**	**Constraints**
dept_name	VarChar	100 characters, primary key, not null
building	VarChar	255 characters
budget	Integer	Unsigned, 0-4294967295

Instructor		
Field	**Data Type**	**Constraints**
id	Integer	Unsigned, 0-4294967295, primary key, not null
name	VarChar	255 characters
dept_name	VarChar	100 characters, foreign key
salary	Integer	Unsigned, 0-4294967295

Advisor		
Field	**Data Type**	**Constraints**
s_id	Integer	Unsigned, 0-4294967295, foreign key, not null
i_id	Integer	Unsigned, 0-4294967295, foreign key, not null

Part 2:

*All the code below was copied from Visual Studio Code, that is why it is formatted in such a way

b, c, d. MySQL statements

```
CREATE TABLE `student`
(
 `ID` int PRIMARY KEY,
 `name` varchar(255),
 `dept_name` varchar(100),
 `tot_cred` smallint
);

CREATE TABLE `takes`
(
 `ID` int NOT NULL,
 `course_id` varchar(255) NOT NULL,
 `sec_id` int NOT NULL,
 `semester` varchar(100) NOT NULL,
 `year` int NOT NULL,
 `grade` varchar(100)
);

CREATE TABLE `section`
(
 `course_id` varchar(255) NOT NULL,
 `sec_id` int NOT NULL,
 `semester` varchar(100) NOT NULL,
 `year` int NOT NULL,
 `building` varchar(255),
 `room_no` int,
 `time_slot_id` varchar(255),
 PRIMARY KEY (`course_id`, `sec_id`, `semester`, `year`)
);

CREATE TABLE `time_slot`
(
 `time_slot_id` varchar(255) NOT NULL,
 `day` varchar(100) NOT NULL,
 `start_hour` int NOT NULL,
 `start_min` int NOT NULL,
 `end_hour` int,
 `end_min` int,
 PRIMARY KEY (`time_slot_id`, `day`, `start_hour`, `start_min`)
);

CREATE TABLE `classroom`
(
 `building` varchar(255),
 `room_no` int,
```

```
 `capacity` int,
 PRIMARY KEY (`building`, `room_no`)
);

CREATE TABLE `course`
(
 `course_id` varchar(255) PRIMARY KEY,
 `title` varchar(255),
 `dept_name` varchar(100),
 `credits` int
);

CREATE TABLE `prereq`
(
 `course_id` varchar(255) NOT NULL,
 `prereq_id` varchar(255) NOT NULL
);

CREATE TABLE `department`
(
 `dept_name` varchar(100) PRIMARY KEY,
 `building` varchar(255),
 `budget` int
);

CREATE TABLE `instructor`
(
 `ID` int PRIMARY KEY,
 `name` varchar(255),
 `dept_name` varchar(100),
 `salary` int
);

CREATE TABLE `teaches`
(
 `ID` int NOT NULL,
 `course_id` varchar(255) NOT NULL,
 `sec_id` int NOT NULL,
 `semester` varchar(100) NOT NULL,
 `year` int NOT NULL,
 PRIMARY KEY (`ID`, `course_id`, `sec_id`, `semester`, `year`)
);

CREATE TABLE `advisor`
(
 `s_id` int NOT NULL,
 `i_id` int NOT NULL
```

```
);


ALTER TABLE `takes` ADD
   FOREIGN KEY (`ID`)
   REFERENCES `student` (`ID`);

ALTER TABLE `takes` ADD
   FOREIGN KEY (`course_id`, `sec_id`, `semester`, `year`)
   REFERENCES `section` (`course_id`, `sec_id`, `semester`, `year`);

ALTER TABLE `section` ADD
   FOREIGN KEY (`building`, `room_no`)
   REFERENCES `classroom` (`building`, `room_no`);

ALTER TABLE `section` ADD
   FOREIGN KEY (`time_slot_id`)
   REFERENCES `time_slot` (`time_slot_id`);

ALTER TABLE `course` ADD
   FOREIGN KEY (`dept_name`)
   REFERENCES `department` (`dept_name`);

ALTER TABLE `prereq` ADD
   FOREIGN KEY (`course_id`)
   REFERENCES `course` (`course_id`);

ALTER TABLE `prereq` ADD
   FOREIGN KEY (`prereq_id`)
   REFERENCES `course` (`course_id`);

ALTER TABLE `teaches` ADD
   FOREIGN KEY (`course_id`, `sec_id`, `semester`, `year`)
   REFERENCES `section` (`course_id`, `sec_id`, `semester`, `year`);

ALTER TABLE `teaches` ADD
   FOREIGN KEY (`ID`)
   REFERENCES `instructor` (`ID`);

ALTER TABLE `instructor` ADD
   FOREIGN KEY (`dept_name`)
   REFERENCES `department` (`dept_name`);

ALTER TABLE `advisor` ADD
   FOREIGN KEY (`s_id`)
   REFERENCES `student` (`ID`);
```

```
ALTER TABLE `advisor` ADD
   FOREIGN KEY (`i_id`)
   REFERENCES `instructor` (`ID`);
```

Part 2:

e. SQL Queries

```
/* Query Number 1 */
SELECT DISTINCT student.name FROM student
   JOIN takes ON student.id = takes.id
   WHERE takes.course_id LIKE 'CS%';

/* Query Number 2 */
SELECT DISTINCT student.name, student.ID FROM student
   JOIN takes ON student.id = takes.id
   WHERE takes.grade LIKE 'F%';

/* Query Number 3 */
SELECT dept_name, max(salary) FROM instructor
   GROUP BY dept_name;

/* Query Number 4 */
SELECT DISTINCT course.title, student.name FROM student
   JOIN takes ON takes.ID = student.ID
   JOIN section ON section.course_id = takes.course_id
   JOIN course ON course.course_id = section.course_id
   JOIN time_slot ON section.time_slot_id = time_slot.time_slot_id
   WHERE time_slot.day LIKE 'F'
   AND time_slot.start_hour > 12
   AND takes.course_id IN
   (SELECT course_id FROM takes
   GROUP BY course_id HAVING count(ID) >= 2 );

/* Query Number 5 */
SELECT DISTINCT instructor.name, teaches.course_id FROM instructor
   JOIN teaches ON teaches.ID = instructor.ID
   JOIN section ON section.course_id = teaches.course_id
   JOIN classroom ON classroom.building = section.building
   AND classroom.room_no = section.room_no
   WHERE classroom.capacity > 50;
```