Application Description

For this project I decided to a model computer system. Each computer system consists of a cpu, cpu cooler, motherboard, ram, power supply, case, storage and optionally a gpu. The database describes systems that have been built and how the components relate to each other.

I've decided to represent each component as an entity and how components interact with each other e.g. connect, as relationships.

This database could be used by a boutique pc systems builder such as origin pc or by a large-scale systems builder such as dell or apple.

The following tables are what I have used reflect the real-world semantics and attributes of a pc system in my database.

The Central_Pocessing_Unit table contains 7 attributes:

- serial_number – this is a unique identifier for that exact cpu.
- motherboard_sn – this is the motherboard that this cpu is linked to.
- name – the model cpu it is e.g. i5 3470k.
- socket – the type of socket the cpu uses, this must match the socket of the motherboard as different sockets have different pin layouts.
- tdp – the thermal design power of the cpu (how much power it dissipates in the form of heat)
- core_count – how many processing cores the cpu has.
- smt – simultaneous multi-threading, can the cpu appear to have more cores than it actually has and as a result process more threads simultaneously.

The CPU cooler table contains 6 attributes:

- serial_number – this is a unique identifier for that exact cpu cooler.
- motherboard_sn – this is the motherboard that this cpu cooler is linked to.
- name – the model cpu cooler it is e.g. corsair h100i
- socket – the type of socket the cpu cooler mounts to, this must match the socket of the motherboard and the cpu the cooler cools.
- fan_rpm – the speed that the cooling fan rotates at (rpm – revolutions per minute).
- noise_level – how loud the cooler is.

The motherboard table contains 6 attributes:

- serial_number – this is a unique identifier for that exact motherboard.
- name – the model motherboard it is e.g. asus prime x370 pro.
- socket – the type of cpu socket the motherboard has e.g. LGA1150
- ram_type – the type of ram that the motherboard accepts e.g. ddr4, ddr3 etc
- ram_slots – the number of ram slots a motherboard has
- form_factor – the type and size of the motherboard, e.g. atx, m-atx etc.

The Graphics_Processing_Unit table contains 6 attributes

- serial_number – this is a unique identifier for that exact gpu.
- motherboard_sn – this is the motherboard that this gpu is linked to.
- name – the model gpu it is e.g. nvidia geforce gtx 980ti
- core_clockk – the speed in mhz at which the graphics card operates e.g. 2000mhz
- length – the length of the gpu, this is important when mounting the gpu in a case.
- memory – the amount of vram (video ram) the gpu has, more vram is desirable for various simulation tasks.

The Random_Access_Memory table contains 6 attributes

- serial_number – this is a unique identifier for that exact cpu cooler.
- motherboard_sn – this is the motherboard that this cpu cooler is linked to.
- name – the model cpu cooler it is e.g. corsair h100i
- number_of_modules – how many sticks of ram the kit contains.
- type – the type of ram e.g. ddr3
- capacity - the amount of space in gb

The Storage_Drive table contains 6 attributes

- serial_number – this is a unique identifier for that exact storage device.
- name – the model of storage device it is e.g. Samsung 850 evo
- case_sn – the case this storage device is mounted in
- capacity – the amount of space in gb that the device has e.g. 500gb
- interface – the type of interface it uses to connect to the rest of the pc e.g. sata, m.2 etc
- form_factor – the size and form of the drive e.g. 2.5inch, 3.5inch, pci-e.

The Computer_Case table contains 5 attributes

- serial_number – this is a unique identifier for that exact case.
- name – the model of case it is e.g. fractal define s.
- storage_bays – the type of storage bays the case supports e.g. 2.5inch, 3.5 inch.
- psu_sn – the power supply that is mounted in this case.
- gpu_space – the amount of space in mm that is available to mount graphics cards.

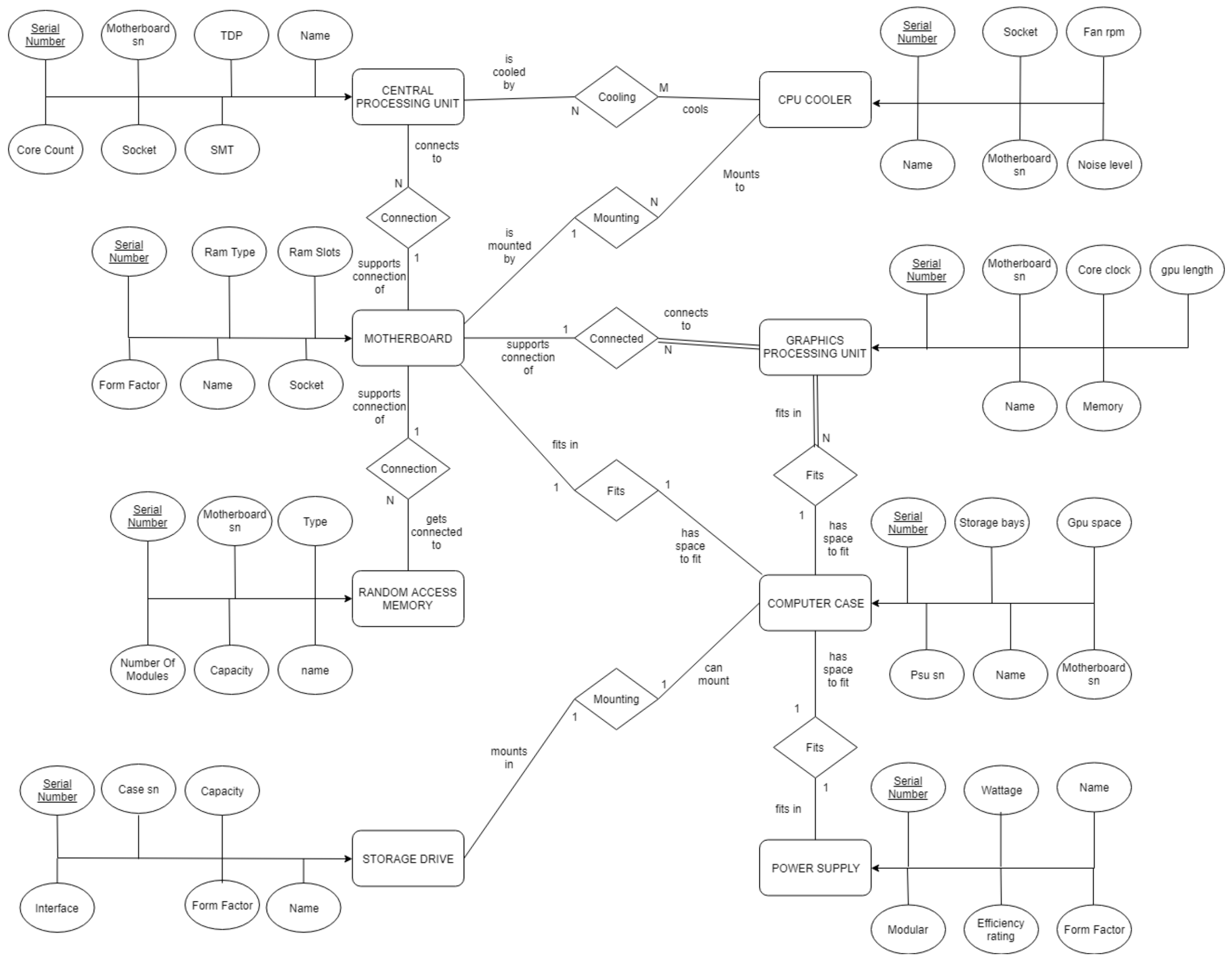
The Power_Supply table contains 6 attributes

- serial_number – this is a unique identifier for that exact power supply.
- name – the model of power supply it is e.g. corsair cs600m.
- wattage – how many watts of power the power supply can deliver e.g. 600 watts
- modular – whether or not the cables an be removed from the power supply.
- efficiency_rating – how efficient the power supply is at converting to different voltages e.g. bronze, gold.
- form_factor – the size of the power supply e.g. atx, itx etc.

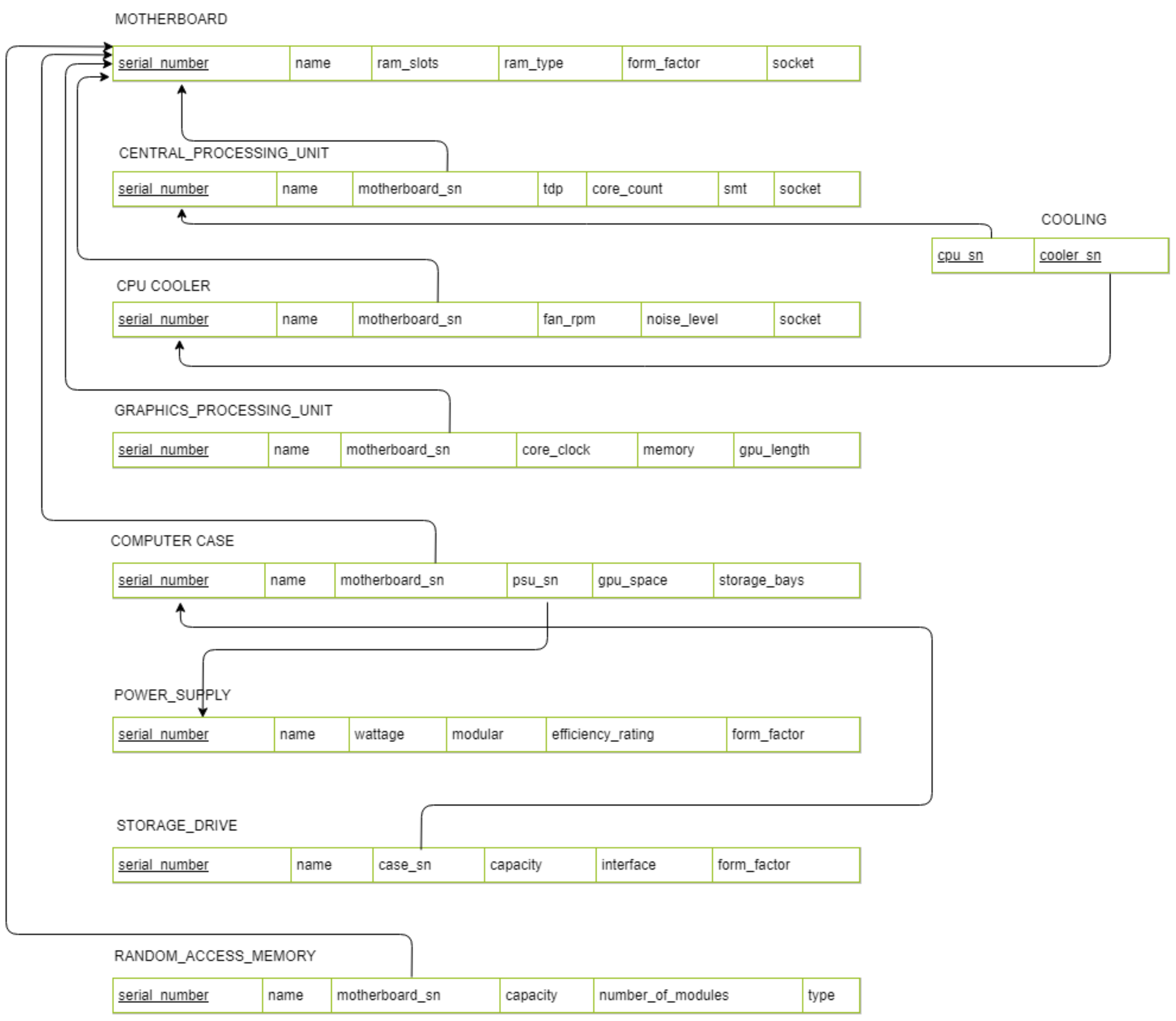
In addition to the assumptions listed above, I have made the following assumptions when making this model:

- Certain cpus require more than one cooler to cool them. E.g. high-performance server hardware, or new experimental unrealised cpus.
- Cases only support one type of power supply e.g. atx.
- Pcs only contain one type of storage.
- Pcs do not require a dedicated gpu as most cpus have onboard graphics, this makes the existence of the gpu dependant on the rest of the system.
- Motherboards can have multiple sockets and therefore house multiple cpus.
- Each component has a unique id tied to it, this id uniquely identifies an instance of that component.
- Components cannot be reused across multiple pcs.
- Pcs that have been released to the public have serial numbers with the prefix “R-“.

Entity Relationship Diagram

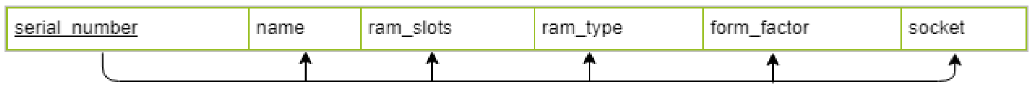


Relational Schema

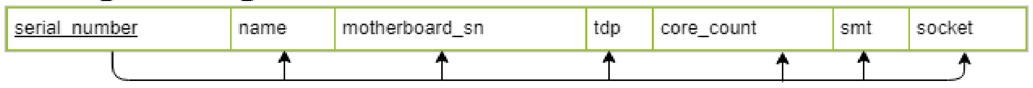

MOTHERBOARD
serial_number | name | ram_slots | ram_type | form_factor | socket
CENTRAL_PROCESSING_UNIT
serial_number | name | motherboard_sn | tdp | core_count | smt | socket
COOLING
cpu_sn | cooler_sn
CPU COOLER
serial_number | name | motherboard_sn | fan_rpm | noise_level | socket
GRAPHICS_PROCESSING_UNIT
serial_number | name | motherboard_sn | core_clock | memory | gpu_length
COMPUTER CASE
serial_number | name | motherboard_sn | psu_sn | gpu_space | storage_bays
POWER_SUPPLY
serial_number | name | wattage | modular | efficiency_rating | form_factor
STORAGE_DRIVE
serial_number | name | case_sn | capacity | interface | form_factor
RANDOM_ACCESS_MEMORY
serial_number | name | motherboard_sn | capacity | number_of_modules | type

Functional Dependency Diagram

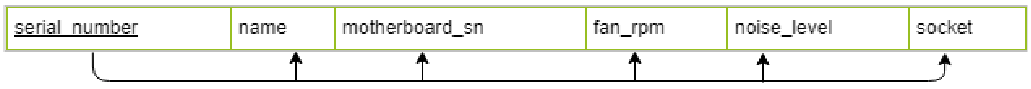
MOTHERBOARD



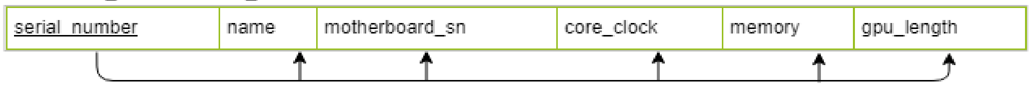
CENTRAL_PROCESSING_UNIT



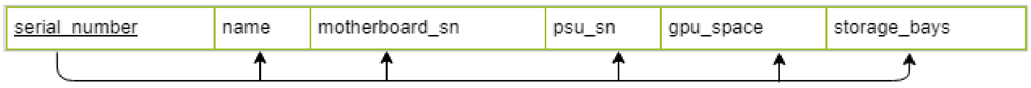
CPU COOLER



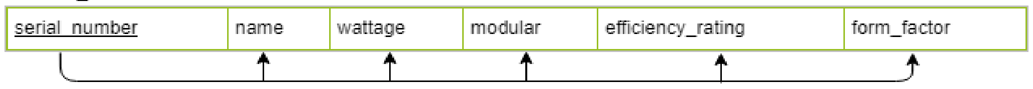
GRAPHICS_PROCESSING_UNIT



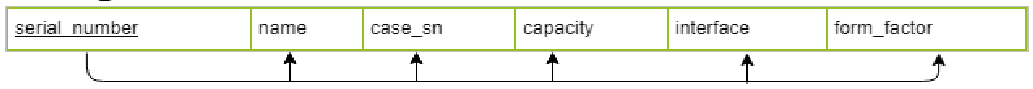
CASE



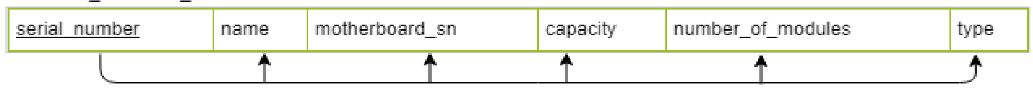
POWER_SUPPLY



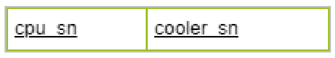
STORAGE_DRIVE



RANDOM_ACCESS_MEMORY



COOLING



Normalisation

After creating the first draft of my entity relationship and functional dependency diagrams, I began to see if any normalisation could be carried out. This is key in the database design process to avoid abnormalities when inserting or deleting values in the database tables. It also ensures the tables will be structured in such a way that they cannot contain redundant data. The rules for normalisation are called normal forms.

First normal form

A relation is in first normal form if the domain of each attribute contains only atomic values and the value of each attribute contains only a single value form that domain.

All relations were found to be in first normal form except for the case relation. Initially I had modelled the case relation with a storage attribute that would contain a list of storage drives that were inserted in the case. To remedy this issue I added a case_sn attribute to the storage relation instead. This way each storage drive is linked to case using a relationship and all attributes of the case relation are now atomic.

Second normal form

A relation is in second normal form if in addition to satisfying the criteria for first normal form, every non-key attribute is fully functionally dependent on the entire primary key.

All relations were found to comply with the rules of second normal form.

Third normal form

A relation is in third normal form if in addition to satisfying the criteria for second normal form, no non-key attributes are transitively dependent upon the primary key.

All relations were found to comply with the rules of second normal form.

Boyce Codd Normal Form (3.5 normal from)

A relation is in Boyce Codd normal form if “all attributes in a relation should be dependent on the key, the while key and nothing but the key”.

All relations were found to comply with the rules of Boyce Codd normal form.

Integrity Constraints

The relational model describes integrity constraints. Three types of integrity constraints considered part of the relational model are:

Key, Entity integrity, Referential integrity.

The dbms must enforce these constraints and we must ensure that our sql operations do not violate any of these constraints

Key Constraints

Key constraints specify that there may not be any duplicate entries in key attributes. Keys are used to uniquely identify a tuple. This will be handled by the dbms for us when we define an attribute as a key.

Entity Integrity Constraints

Entity constraints are specified on individual relations, one such constraint is that none of the primary key may be null. In my case, entity constraints are handled by the dbms.

Referential Integrity Constraints

Referential integrity constraints are specified between two relations – they maintain consistency among tuples in the two relations

A tuple in one relation that refers to another relation, must refer to an existing tuple in that relation. A foreign key formally specifies a referential integrity constrain between two relations, for this we must be careful with our sql operations and define when foreign keys may or may not be null. E.g. we can only insert a cpu into the cpu table if it is linked to a motherboard that already exists in the motherboard table.

Semantic Constraints

Semantic constraints are vital to ensure the integrity of the database and avoid the accidental corruption of information. These constraints can be defined when creating the tables in the database and in the form of checks, triggers and assertions. For my semantic constraints I decided to implement a trigger for parts that are inserted with no name. Rather than leave that attribute of the table blank, update its value to “undecided”, as any parts inserted without a name are most likely still in production and unreleased.

```
#Create semantic constraints
#Trigger will set the names of inserted cpus that are null
CREATE TRIGGER set_default_cpu_name BEFORE UPDATE ON central_processing_unit
     FOR EACH ROW
     BEGIN
         IF NEW.name IS NULL THEN
             SET NEW.name = "undecided";
         END IF;
    END;
```

Security Constraints

Security constraints are concerned with deliberate corruption of data in the database. We can ensure database security using security policies and access control.

For security I have implemented separate views for every table. These views only contain products which have been released to the public. This would be useful when defining different user privilege level. I this system was being used by both staff and customers, the staff may have access to the complete tables whereas customers will on get access to the restricted views that contain released products.

This was achieved by creating different roles, the staff role have full permissions to all the tables and views in the database, including grant option as some business partners may need access to unreleased products. The customer role then has only read permissions on the restricted views created for each component.

```sql
#Create roles for different access levels
CREATE ROLE staff;
CREATE ROLE customer;

#Set permissions for staff
GRANT ALL ON motherboard TO staff;
GRANT ALL ON central_processing_unit TO staff;
GRANT ALL ON cpu_cooler TO staff;
GRANT ALL ON graphics_processing_unit TO staff;
GRANT ALL ON power_supply TO staff;
GRANT ALL ON computer_case TO staff;
GRANT ALL ON storage_drive TO staff;
GRANT ALL ON random_access_memory TO staff;
GRANT ALL ON cooling TO staff;
GRANT ALL ON motherboard_released TO staff;
GRANT ALL ON central_processing_unit_released TO staff;
GRANT ALL ON cpu_cooler_released TO staff;
GRANT ALL ON graphics_processing_unit_released TO staff;
GRANT ALL ON power_supply_released TO staff;
GRANT ALL ON computer_case_released TO staff;
GRANT ALL ON storage_drive_released TO staff;
GRANT ALL ON random_access_memory_released TO staff;

#Set permissions for customer
GRANT SELECT ON motherboard_released TO customer;
GRANT SELECT ON central_processing_unit_released TO customer;
GRANT SELECT ON cpu_cooler_released TO customer;
GRANT SELECT ON graphics_processing_unit_released TO customer;
GRANT SELECT ON power_supply_released TO customer;
GRANT SELECT ON computer_case_released TO customer;
GRANT SELECT ON storage_drive_released TO customer;
GRANT SELECT ON random_access_memory_released TO customer;
```

Appendix:

```sql
#An sql database that represents computers and how their parts fit together

DROP DATABASE IF EXISTS computerparts;
CREATE DATABASE computerparts;
USE computerparts;

#Define tables and constraints
CREATE TABLE motherboard (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    ram_slots int NOT NULL CHECK (ram_slots > 0),
    ram_type varchar(4) NOT NULL,
    form_factor varchar(5) NOT NULL,
    socket varchar(10) NOT NULL,
    PRIMARY KEY (serial_number)
    );

CREATE TABLE central_processing_unit (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    motherboard_sn varchar(12) NOT NULL,
    tdp int NOT NULL,
    core_count int NOT NULL CHECK (core_count > 0),
    smt varchar(3) NOT NULL,
    socket varchar(10) NOT NULL,
    PRIMARY KEY (serial_number),
    FOREIGN KEY (motherboard_sn)
    REFERENCES motherboard(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
  );

CREATE TABLE cpu_cooler (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    motherboard_sn varchar(12) NOT NULL,
    fan_rpm int NOT NULL,
    noise_level int NOT NULL,
    socket varchar(10) NOT NULL,
    PRIMARY KEY (serial_number),
    FOREIGN KEY (motherboard_sn)
    REFERENCES motherboard(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );

CREATE TABLE graphics_processing_unit (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    motherboard_sn varchar(12) NOT NULL,
    core_clock int NOT NULL,
    memory int NOT NULL,
    gpu_length int NOT NULL CHECK (gpu_length > 0),
    PRIMARY KEY (serial_number),
    FOREIGN KEY (motherboard_sn)
    REFERENCES motherboard(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );

CREATE TABLE power_supply (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    wattage int NOT NULL,
    modular varchar(3) NOT NULL,
    efficiency_rating varchar(8) NOT NULL,
    form_factor varchar(5) NOT NULL,
    PRIMARY KEY (serial_number)
    );

CREATE TABLE computer_case (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    motherboard_sn varchar(12) NOT NULL,
    psu_sn varchar(12) NOT NULL,
    gpu_space int NOT NULL,
    storage_bays varchar(7) NOT NULL,
    PRIMARY KEY (serial_number),
    FOREIGN KEY (motherboard_sn)
    REFERENCES motherboard(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
    FOREIGN KEY (psu_sn)
    REFERENCES power_supply(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );

CREATE TABLE storage_drive (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    case_sn varchar(12) NOT NULL,
    capacity int NOT NULL CHECK (capacity > 0),
    interface varchar(4) NOT NULL,
    form_factor varchar(7) NOT NULL,
```

```sql
    PRIMARY KEY (serial_number),
    FOREIGN KEY (case_sn)
    REFERENCES computer_case(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );

CREATE TABLE random_access_memory (
    serial_number varchar(12) NOT NULL,
    name varchar(50),
    motherboard_sn varchar(12) NOT NULL,
    capacity int NOT NULL,
    number_of_modules int NOT NULL CHECK (number_of_modules > 0),
    type varchar(4) NOT NULL,
    PRIMARY KEY (serial_number),
    FOREIGN KEY (motherboard_sn)
    REFERENCES motherboard(serial_number)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );

CREATE TABLE cooling (
    cpu_sn varchar(12) NOT NULL,
    cooler_sn varchar(12) NOT NULL,
    PRIMARY KEY (cpu_sn, cooler_sn));

#Create restricted views for security and different access levels
CREATE VIEW motherboard_released
AS SELECT serial_number, name , ram_slots, ram_type, form_factor, socket
FROM motherboard
WHERE serial_number LIKE "R-%";

CREATE VIEW central_processing_unit_released
AS SELECT serial_number, name, motherboard_sn, tdp, core_count, smt, socket
FROM central_processing_unit
WHERE serial_number LIKE "R-%"
AND motherboard_sn LIKE "R-%";

CREATE VIEW cpu_cooler_released
AS SELECT serial_number, name, motherboard_sn, fan_rpm, noise_level, socket
FROM cpu_cooler
WHERE serial_number LIKE "R-%"
AND motherboard_sn LIKE "R-%";

CREATE VIEW graphics_processing_unit_released
AS SELECT serial_number, name, motherboard_sn, core_clock, memory, gpu_length
FROM graphics_processing_unit
WHERE serial_number LIKE "R-%"
```

```sql
AND motherboard_sn LIKE "R-%";

CREATE VIEW power_supply_released
AS SELECT serial_number, name, wattage, modular, efficiency_rating, form_facto
r
FROM power_supply
WHERE serial_number LIKE "R-%";

CREATE VIEW computer_case_released
AS SELECT serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays

FROM computer_case
WHERE serial_number LIKE "R-%"
AND motherboard_sn LIKE "R-%"
AND psu_sn LIKE "R-%";

CREATE VIEW storage_drive_released
AS SELECT serial_number, name, case_sn, capacity, interface, form_factor
FROM storage_drive
WHERE serial_number LIKE "R-%";

CREATE VIEW random_access_memory_released
AS SELECT serial_number, name, motherboard_sn, capacity, number_of_modules, ty
pe
FROM random_access_memory
WHERE serial_number LIKE "R-%"
AND motherboard_sn LIKE "R-%";

#Create roles for different access levels
CREATE ROLE staff;
CREATE ROLE customer;

#Set permissions for staff
GRANT ALL ON motherboard TO staff;
GRANT ALL ON central_processing_unit TO staff;
GRANT ALL ON cpu_cooler TO staff;
GRANT ALL ON graphics_processing_unit TO staff;
GRANT ALL ON power_supply TO staff;
GRANT ALL ON computer_case TO staff;
GRANT ALL ON storage_drive TO staff;
GRANT ALL ON random_access_memory TO staff;
GRANT ALL ON cooling TO staff;
GRANT ALL ON motherboard_released TO staff;
GRANT ALL ON central_processing_unit_released TO staff;
GRANT ALL ON cpu_cooler_released TO staff;
GRANT ALL ON graphics_processing_unit_released TO staff;
GRANT ALL ON power_supply_released TO staff;
GRANT ALL ON computer_case_released TO staff;
GRANT ALL ON storage_drive_released TO staff;
GRANT ALL ON random_access_memory_released TO staff;

#Set permissions for customer
GRANT SELECT ON motherboard_released TO customer;
GRANT SELECT ON central_processing_unit_released TO customer;
GRANT SELECT ON cpu_cooler_released TO customer;
GRANT SELECT ON graphics_processing_unit_released TO customer;
GRANT SELECT ON power_supply_released TO customer;
GRANT SELECT ON computer_case_released TO customer;
GRANT SELECT ON storage_drive_released TO customer;
GRANT SELECT ON random_access_memory_released TO customer;

#Create semantic constraints
#Trigger will set the names of inserted cpus that are null
CREATE TRIGGER set_default_cpu_name BEFORE UPDATE ON central_processing_unit
     FOR EACH ROW
     BEGIN
         IF NEW.name IS NULL THEN
             SET NEW.name = "undecided";
         END IF;
    END;



#Insert entries into the motherboard table
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("R-M123456781", "Asus Prime x370", 4, "ddr4", "atx", "am4");
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("R-M123456782", "Gigabyte ds3h", 4, "ddr3", "atx", "lga1155");
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("R-M123456783", "Asus x99-pro", 8, "ddr4", "atx", "lga2011v3");
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("R-M123456784", "Aus ROG Zenith Extreme", 8, "ddr4", "e-
atx", "tr4");
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("R-M123456785", "Aorus pro ac wifi ", 2, "ddr4", "m-itx", "am4");
INSERT INTO motherboard (
    serial_number, name , ram_slots, ram_type, form_factor, socket)
    VALUES ("U-M123456786", "Project X dual socket board", 8, "ddr4", "e-
atx", "lga2066");

#Insert entries into the cpu table
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("R-C123456781", "Amd Ryzen 5 1600", "R-
M123456781", 65, 6, "yes", "am4");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("R-C123456782", "Intel Core i5 3570k", "R-
M123456782", 77, 4, "no", "lga1155");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("R-C123456783", "Intel Core i7 5820k", "R-
M123456783", 140, 6, "yes", "lga2011v3");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("R-C123456784", "Amd Threadripper 3970X", "R-
M123456784", 280 , 32, "yes", "tr4");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("R-C123456785", "Amd Ryzen 7 1800x", "R-
M123456785", 95, 8, "yes", "am4");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("U-C123456786", "Cascade Lake-AP", "U-
M123456786", 200, 64, "yes", "lga2066");
INSERT INTO central_processing_unit (
    serial_number, name, motherboard_sn, tdp, core_count, smt, socket)
    VALUES ("U-C123456787", "Cascade Lake-AP", "U-
M123456786", 200, 64, "yes", "lga2066");


#Insert entries into the cpu cooler table
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("R-CC12345671", "Corsair h100i", "R-M123456781", 2200, 20, "am4");
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("R-CC12345672", "Intel Stock Cooler", "R-
M123456782", 1200, 30, "lga1155");
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("R-CC12345673", "EK Supremecy Evo", "R-
M123456783", 2200, 15, "lga2011v3");
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("R-CC12345674", "Dual stack noctua cooler", "R-
M123456784", 1800, 35, "tr4");
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("R-CC12345675", "NZXT Kraken", "R-M123456785", 2200, 20, "am4");
INSERT INTO cpu_cooler (
```

```sql
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("U-CC1234567x", "LN2 pt1", "U-M123456786", 1000, 5, "lga2066");
INSERT INTO cpu_cooler (
    serial_number, name, motherboard_sn, fan_rpm, noise_level, socket)
    VALUES ("U-CC1234567y", "LN2 pt2", "U-M123456786", 1000, 5, "lga2066");

    #Insert entries into the gpu table
INSERT INTO graphics_processing_unit (
    serial_number, name, motherboard_sn, core_clock, memory, gpu_length)
    VALUES ("R-G123456781", "Asus Geforce Gtx 970", "R-
M123456781", 1114, 4, 280);
INSERT INTO graphics_processing_unit (
    serial_number, name, motherboard_sn, core_clock, memory, gpu_length)
    VALUES ("R-G123456782", "Gigabyte Geforce Gtx 650ti", "R-
M123456782", 928, 1, 144);
INSERT INTO graphics_processing_unit (
    serial_number, name, motherboard_sn, core_clock, memory, gpu_length)
    VALUES ("R-G123456783", "Nvidia Quadro GV100", "R-
M123456783", 1132, 32, 266);
INSERT INTO graphics_processing_unit (
    serial_number, name, motherboard_sn, core_clock, memory, gpu_length)
    VALUES ("R-G123456784", "Amd Radeon VII", "R-M123456784", 1400, 16, 305);
INSERT INTO graphics_processing_unit (
    serial_number, name, motherboard_sn, core_clock, memory, gpu_length)
    VALUES ("R-G123456785", "Amd r9 290x", "R-M123456785", 1000, 4, 276);

#Insert entries into the psu table
INSERT INTO power_supply (
    serial_number, name, wattage, modular, efficiency_rating, form_factor)
    VALUES ("R-P123456781", "Corsair cx600m", 600, "yes", "bronze", "atx");
INSERT INTO power_supply (
    serial_number, name, wattage, modular, efficiency_rating, form_factor)
    VALUES ("R-P123456782", "Evga br500", 500, "no", "bronze", "atx");
INSERT INTO power_supply (
    serial_number, name, wattage, modular, efficiency_rating, form_factor)
    VALUES ("R-P123456783", "Corsair rm850x", 850, "yes", "gold", "atx");
INSERT INTO power_supply (
    serial_number, name, wattage, modular, efficiency_rating, form_factor)
    VALUES ("R-
P123456784", "Corsair ax1200i", 1200, "yes", "platinum", "atx");
INSERT INTO power_supply (
    serial_number, name, wattage, modular, efficiency_rating, form_factor)
    VALUES ("R-P123456785", "Silverstone SX600G", 600, "yes", "gold", "m-
itx");

#Insert entries into the case table
INSERT INTO computer_case (
    serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays)
    VALUES ("R-CE12345671", "Fractal Design Define S", "R-M123456781", "R-
P123456781", "400", "2.5inch");
INSERT INTO computer_case (
    serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays)
    VALUES ("R-CE12345672", "Fractal Design Define R5", "R-M123456782", "R-
P123456782", "310", "3.5inch");
INSERT INTO computer_case (
    serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays)
    VALUES ("R-CE12345673", "Zalman z11 plus", "R-M123456783", "R-
P123456783", "290", "3.5inch");
INSERT INTO computer_case (
    serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays)
    VALUES ("R-CE12345674", "Nzxt h440", "R-M123456784", "R-
P123456784", "294", "3.5inch");
INSERT INTO computer_case (
    serial_number, name, motherboard_sn, psu_sn, gpu_space, storage_bays)
    VALUES ("R-CE12345675", "Ncase M1", "R-M123456785", "R-
P123456785", "280", "2.5inch");


#Insert entries into the storage table
INSERT INTO storage_drive (
    serial_number, name, case_sn, capacity, interface, form_factor)
    VALUES ("R-S123456781", "Samsung 850 Evo", "R-
CE12345671", 500, "sata", "2.5inch");
INSERT INTO storage_drive (
    serial_number, name, case_sn, capacity, interface, form_factor)
    VALUES ("R-S123456782", "Western Digital Caviar Green 1tb", "R-
CE12345672", 1000, "sata", "3.5inch");
INSERT INTO storage_drive (
    serial_number, name, case_sn, capacity, interface, form_factor)
    VALUES ("R-S123456783", "Seagate Barracuda 2tb", "R-
CE12345673", 2000, "sata", "3.5inch");
INSERT INTO storage_drive (
    serial_number, name, case_sn, capacity, interface, form_factor)
    VALUES ("R-S123456784", "Samsung HD105SI", "R-
CE12345674", 1000, "sata", "3.5inch");
INSERT INTO storage_drive (
    serial_number, name, case_sn, capacity, interface, form_factor)
    VALUES ("R-S123456785", "Intel 730", "R-
CE12345675", 240, "sata", "2.5inch");

#Insert entries into ram table
INSERT INTO random_access_memory (
    serial_number, name, motherboard_sn, capacity, number_of_modules, type)
    VALUES ("R-R123456781", "Corsair Vengeance LPX", "R-
M123456781", 16, 2, "ddr4");
INSERT INTO random_access_memory (
    serial_number, name, motherboard_sn, capacity, number_of_modules, type)
    VALUES ("R-R123456782", "G Skill Ripjaws", "R-M123456782", 8, 1, "ddr3");
INSERT INTO random_access_memory (
    serial_number, name, motherboard_sn, capacity, number_of_modules, type)
    VALUES ("R-R123456783", "Corsair Vengeance RGB", "R-
M123456783", 16, 2, "ddr4");
INSERT INTO random_access_memory (
    serial_number, name, motherboard_sn, capacity, number_of_modules, type)
    VALUES ("R-R123456784", "Corsair Vengeance LPX", "R-
M123456784", 128, 8, "ddr4");
INSERT INTO random_access_memory (
    serial_number, name, motherboard_sn, capacity, number_of_modules, type)
    VALUES ("R-R123456785", "Crucial Ballistix LT", "R-
M123456785", 32, 2, "ddr4");

#Insert enties into the cooling table to link cpus and coolers together
INSERT INTO cooling (cpu_sn, cooler_sn)
    VALUES ("R-C123456781", "R-M123456781");
INSERT INTO cooling (cpu_sn, cooler_sn)
    VALUES ("R-C123456782", "R-M123456782");
INSERT INTO cooling (cpu_sn, cooler_sn)
    VALUES ("R-C123456783", "R-M123456783");
INSERT INTO cooling (cpu_sn, cooler_sn)
    VALUES ("R-C123456784", "R-M123456784");
INSERT INTO cooling (cpu_sn, cooler_sn)
    VALUES ("R-C123456785", "R-M123456785");
```