Jonathon M. Thompson
CS. 355

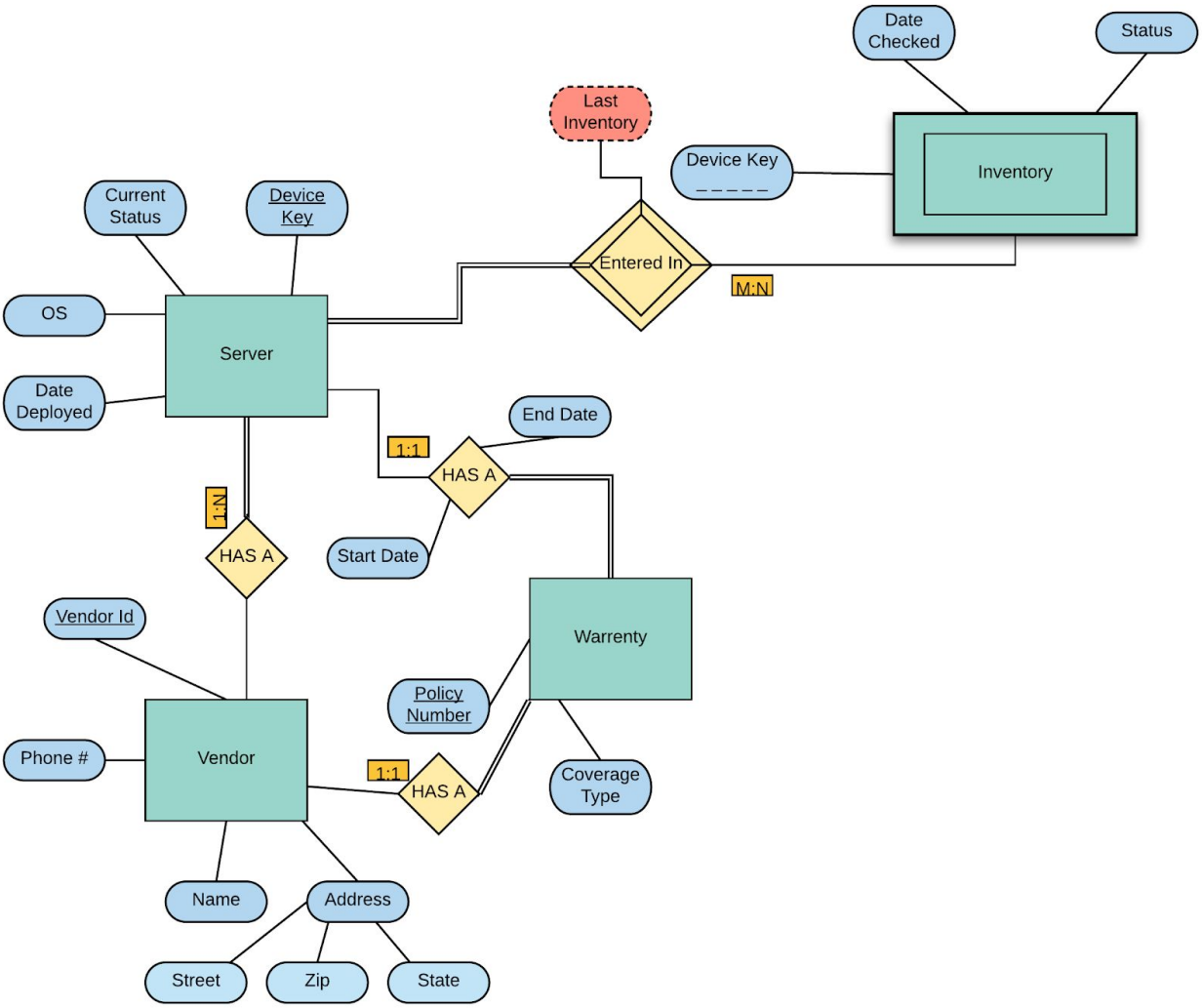
Inventory Database

The goal of this database is to manage and maintain a server inventory. This database has been set up for a data center who deploys different servers from various vendors.

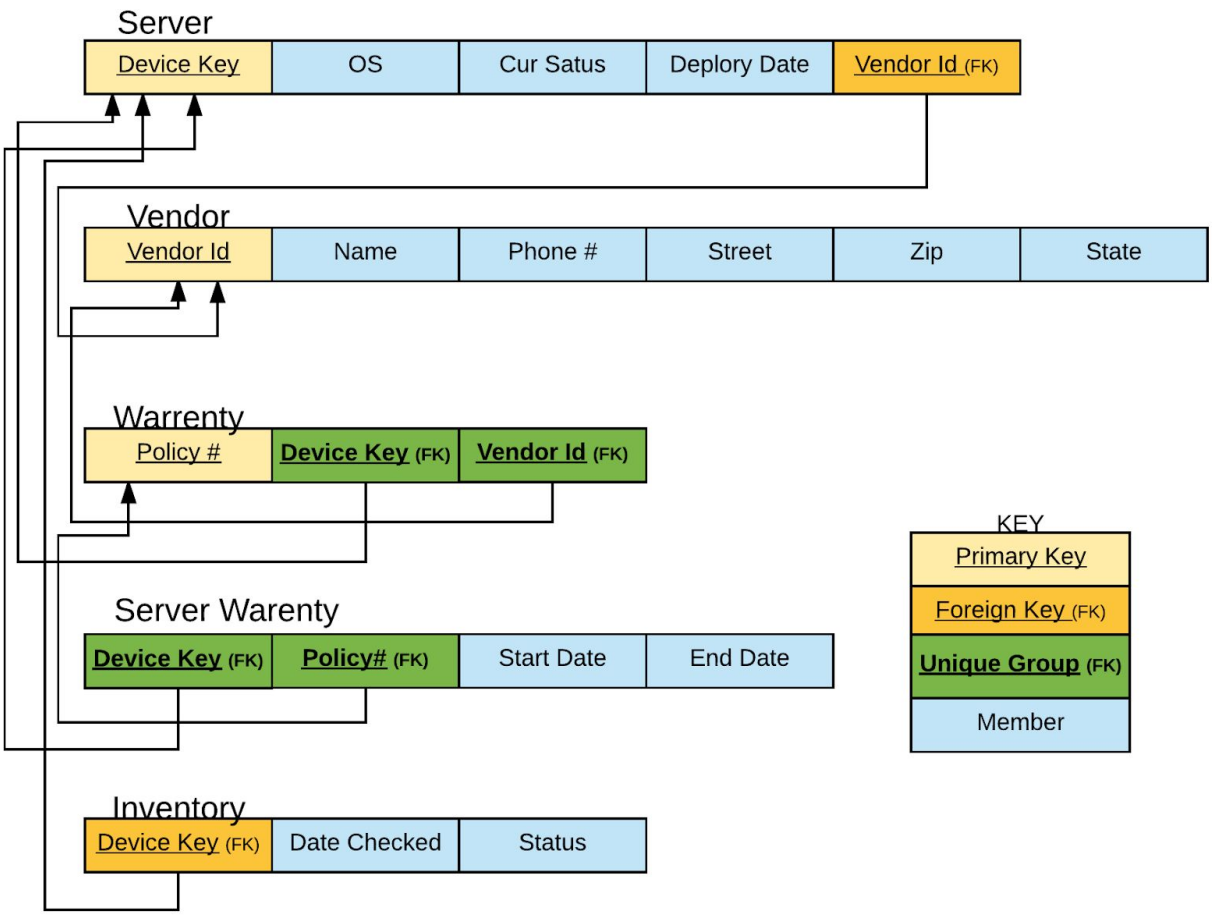
The data center is responsible to a greater entity who cares deeply about their investment and as such the data center does annual walk throughs to re-inventory their servers while also monitoring the current status of any given server so they can service it promptly if need be. During the annual inventory the current status of the servers are logged for long term tracking. They also keep track of the vendors who sold the servers to them including all necessary contact information so they can call for help or parts. Another crucial part of this is the warranty information maintained for unforeseen failures or catastrophes.

The Business logic is as follows. The data center appropriated funds for new servers. They purchase a server and a warranty to go with it at which time they are assigned a policy for. Upon receiving the server they load it into the server table, and record it in the warranty table linking the server to policy. If the vendor is in the list of vendors that vendor is noted alongside the server. A program does regular scans of the servers once they have been deployed and updates that server's current status in the database. As previously noted, an annual walkthrough is done recording the current state of all servers still in operation for the purpose of reporting to those who paid for the servers. In the event a server goes down it will be manually marked as in service while a technician repairs it. If that employee runs into trouble they may contact the vendor for help and reference the listed policy number if need be. Should that server be unrepairable the vendor's address is stored locally for rapid returns.

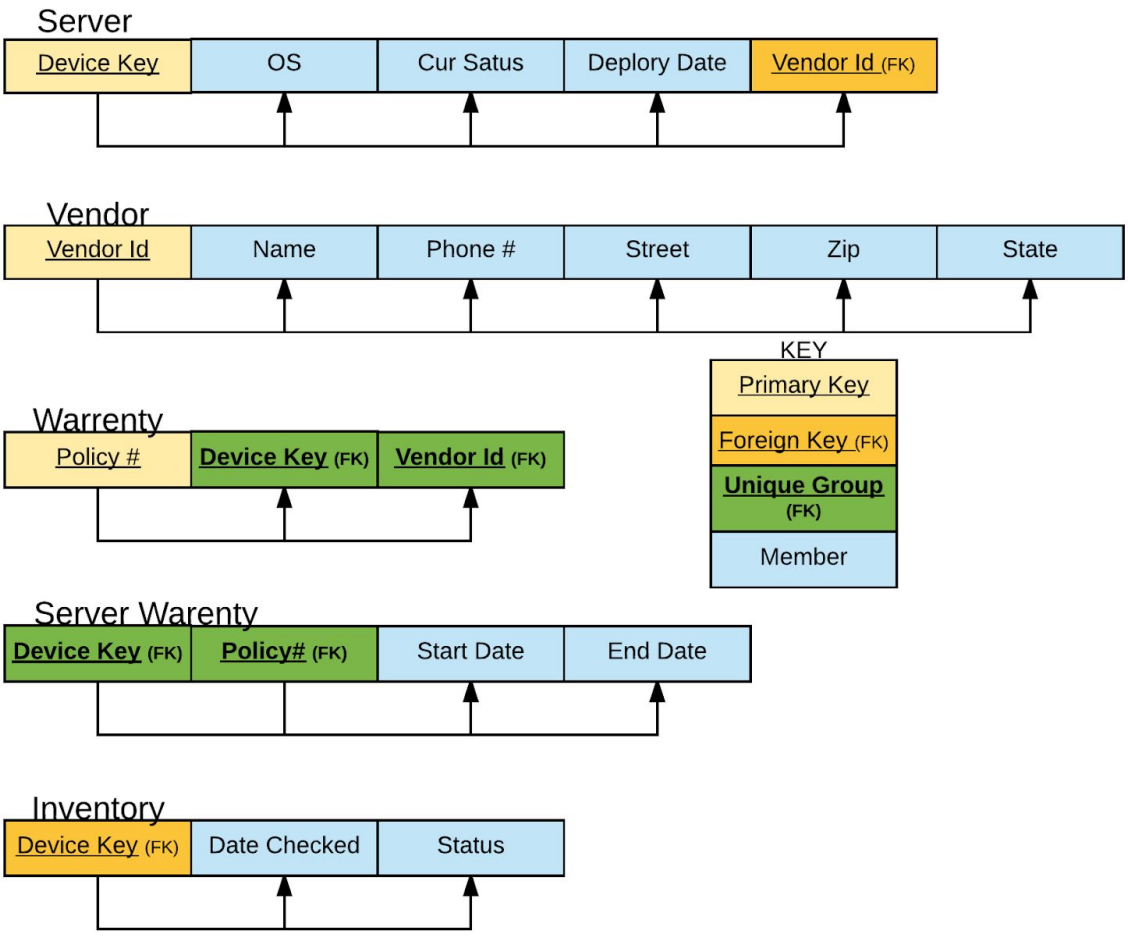
Entity Relationship Diagram



Relational Schema Diagram



Functional Dependency Diagram



Description of Tables

Table 1: Vendor
Purpose: Track vendors
Attributes: Name, Phone, Street, Zip, State
Keys: vendor_id is a primary key
MySql: CREATE TABLE vendor (vendor_id bigint auto_increment primary key, name varchar(30), phone_number bigint, street varchar(30), zip varchar(5), state varchar(15));

vendor_id	name	phone	street	zip	state
1	HP	8005552327	2121 Whisky St	90210	California
2	Dell	8005553145	666 Bourbon St.	35672	Florida
3	Lenovo	8005551234	1821 Scotch Dr.	42127	New York
4	Cisco	8005556666	7777 Crack Head Circle	97217	California
5	Asus	8005559898	1234 Vodka Ct.	07421	Maine

Table 2: Server
Purpose: Track servers
Attributes: OS, Current Status, date_deployed
Keys: device_key is a primary key, vendor_id is a foreign key
MySql: CREATE TABLE server (device_key bigint auto_increment primary key, current_status varchar(25), OS varchar(50), date_deployed datetime, vendor_id bigint, FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id) ON DELETE CASCADE);

device_key	current_status	os	date_deployed	vendor_id
1	Good	Windows Server 2012	2012-12-05 00:00:00	1
2	In Service	Windows Server 2008	2009-01-01 00:00:00	2
3	Good	Windows Server 2008	2009-01-01 00:00:00	2
4	Good	Windows Server 2012	2013-03-03 00:00:00	4
5	Good	Windows Server 2008	2010-01-01 00:00:00	1
6	Good	Windows Server 2012	2014-01-01 00:00:00	2

Table 3: Warranty
Purpose: Track policies between server and vendor
Attributes: policy_id, device_key, vendor_id
Keys: policy_id is a primary key, vendor_id and device_key are foreign keys and together unique
MySql: CREATE TABLE warranty (policy_id bigint auto_increment primary key, device_key bigint, vendor_id bigint,
unique(device_key, vendor_id),
FOREIGN KEY (device_key) REFERENCES server(device_key) ON DELETE CASCADE,
FOREIGN KEY (vendor_id) REFERENCES vendor(vendor_id) ON DELETE CASCADE);

policy_id	device_key	vendor_id
1	1	1
2	2	2
3	3	2
4	4	4
5	5	1
6	6	2

Table 4: server_warranty
Purpose: Track policies coverage time
Attributes: start_date, end_date
Keys: policy_id and device_key are primary keys, unique and foreign
MySql: CREATE TABLE server_warranty (device_key bigint, policy_id bigint, primary key(device_key, policy_id),
unique(device_key, policy_id), start_date datetime, end_date datetime,
FOREIGN KEY (device_key) REFERENCES server(device_key) ON DELETE CASCADE,
FOREIGN KEY (policy_id) REFERENCES warranty(policy_id) ON DELETE CASCADE);

device_key	policy_id	start_date	end_date
1	1	2012-12-01 00:00:00	2016-12-01 00:00:00
2	2	2009-01-01 00:00:00	2013-01-01 00:00:00
3	3	2009-01-01 00:00:00	2013-01-01 00:00:00
4	4	2013-03-03 00:00:00	2017-03-03 00:00:00
5	5	2010-01-01 00:00:00	2014-01-01 00:00:00
6	6	2014-01-01 00:00:00	2018-01-01 00:00:00

Table 5: inventory
Purpose: Track inventory and condition at time of inventory.
Attributes: date_checked, status
Keys: device_key is a primary and foreign key
MySql: CREATE TABLE inventory (device_key bigint, date_checked datetime, status varchar(25), FOREIGN KEY (device_key) REFERENCES server(device_key) ON DELETE CASCADE);

device_key	date_checked	status
1	2013-05-05 00:00:00	Good
2	2013-05-05 00:00:00	In service
3	2013-05-05 00:00:00	Good
4	2013-05-05 00:00:00	Down
5	2013-05-05 00:00:00	Good
6	2013-05-05 00:00:00	Good
1	2014-05-05 00:00:00	Down
2	2014-05-05 00:00:00	Good
3	2014-05-05 00:00:00	Good
4	2014-05-05 00:00:00	Good
5	2014-05-05 00:00:00	Good
6	2014-05-05 00:00:00	In service
1	2015-05-05 00:00:00	Good
2	2015-05-05 00:00:00	Good
3	2015-05-05 00:00:00	Good
4	2015-05-05 00:00:00	In service
5	2015-05-05 00:00:00	Good
6	2015-05-05 00:00:00	In service
1	2016-05-05 00:00:00	Good
2	2016-05-05 00:00:00	Good
3	2016-05-05 00:00:00	In service
4	2016-05-05 00:00:00	In service
5	2016-05-05 00:00:00	Good
6	2016-05-05 00:00:00	Good

Queries

Query 1: Down Servers over time

Reason: It's crucial to monitor the history of servers down during inventory. As a history builds vendors can be contacted with all relevant data present to negotiate future purchases.

MySQL:
```
SELECT i.device_key, status, OS, date_deployed, date_checked, policy_id, name, phone_number, street, zip, state
FROM inventory i
JOIN server s ON s.device_key = i.device_key
JOIN vendor v ON v.vendor_id = s.vendor_id
JOIN warranty w ON w.device_key = i.device_key
WHERE status = 'Down'
GROUP BY i.device_key;
```

device_key	status	os	date_deployed	date_checked	policy_id	name	phone	street	zip	state
1	Down	Windows Server 2012	2012-12-05 00:00:00	2014-05-05 00:00:00	1	HP	8005552327	2121 Whisky St	90210	California
4	Down	Windows Server 2012	2013-03-03 00:00:00	2013-05-05 00:00:00	4	Cisco	8005556666	7777 Crack Head Circle	97217	California

Query 2: Common Vendors

Reason: Is good to be able to look back into the history of servers and return those who were purchased from multiple times.

MySQL:
```
SELECT name, phone_number, count(s.vendor_id) as Purchase_Count
FROM server s
JOIN vendor v ON s.vendor_id = v.vendor_id
GROUP BY s.vendor_id
HAVING count(s.vendor_id) > 1;
```

name	phone_number	Purchase_Count
HP	8005552327	2
Dell	8005553145	3

Query 3: Known Statuses

Reason: Gathering history of all statuses ever recorded

MySQL:
```
SELECT current_status as stat FROM server
UNION
SELECT status as stat FROM inventory;
```

status
Good
In Service
Down

Query 3: Known Statuses in Inventory

Reason: Gathering history of all statuses ever recorded in the inventory
MySQL: SELECT DISTINCT status FROM inventory;

status
Good
In Service
Down

Query 4: Update Status
Reason: It will be necessary to have a query available to update the status of a particular server
MySQL: UPDATE server
SET current_status='Good'
WHERE device_key=4;
Prev:

device_key	current_status
1	Good
2	In Service
3	Good
4	Down
5	Good
6	Good

Now:

device_key	current_status
1	Good
2	In Service
3	Good
4	Good
5	Good
6	Good

Query 5: Latest Inventory Status
Reason: Getting the statuses from the most recent inventory run for rapid reporting could be crucial
MySQL: CREATE OR REPLACE VIEW latest_inventory_status AS
SELECT device_key, status, date_checked
FROM inventory
ORDER BY date_checked DESC
LIMIT 6;

SELECT * FROM latest_inventory_status ORDER BY device_key:

device_key	status	date_checked
1	Good	2016-05-05 00:00:00
2	Good	2016-05-05 00:00:00
3	In service	2016-05-05 00:00:00
4	In service	2016-05-05 00:00:00
5	Good	2016-05-05 00:00:00
6	Good	2016-05-05 00:00:00

Query 6: Status Count in Inventory for given status
Reason: Gathering the number of good, down, or in service is necessary for rapid reporting
MySQL: CREATE OR REPLACE VIEW status_counts AS
SELECT device_key, COUNT(device_key) AS count, status FROM inventory
GROUP BY status;

DROP FUNCTION IF EXISTS stat_count;

DELIMITER //
CREATE FUNCTION stat_count(_status varchar(25)) RETURNS bigint
BEGIN
DECLARE theCount bigint;

SELECT count INTO theCount
FROM status_counts
WHERE status = _status;

RETURN theCount;

END//
DELIMITER ;

SELECT stat_count('Good');

stat_count(‘Good’)
16

Query 7: Rapid Response
Reason: getting vendor phone numbers for servers that go down is going to be very important for rapid response
MySQL:

```
CREATE OR REPLACE VIEW reportView as
SELECT s.device_key, current_status, name, phone_number
FROM server s
JOIN vendor v ON v.vendor_id = s.vendor_id;

DROP PROCEDURE IF EXISTS d;

DELIMITER //
CREATE PROCEDURE d (_current_status varchar(255))
        BEGIN
SELECT device_key, name, phone_number
FROM reportView
WHERE current_status = _current_status;
        END //
DELIMITER ;
```

CALL d('Down');

device_key	name	phone_number
4	Cisco	8005556666