Key Tracker

Final Project

Eunbi Kim, Elizabeth (Liz) Myers, Nehal Taya, Rutu Hasmukh Waghela

The Scenario

Why do we think about the database system for tracking keys?

The Rescue Mission provides emergency housing, meals and services to the unhoused in Syracuse, NY.

- 400 employees
- 11 housing, admin building, 27 stores and donation centers
- 700 distinct keys
- (Problem) The current key tracking software is on a PC from the mid-90's, using an Alpha Four database.

Requirements

What functions do we need from the key tracking database?

Function 1

Viewing what keys are assigned to an employee.

Use case: HR looks up employee to verify what keys should be returned when off boarding staff.

Function 2

Viewing all employees that are assigned a particular key.

Use case: When rekeying a door lock, Facilities can look up everyone who has access to that lock to make sure they get the new key.

Function 3

Track key data (storage hook, bitting, blank type, keys on hand)

Use case: The information facilities needs to make keys (blank, bitting, stamp) needs to be easily available.

Function 4

Track asset data

Use case: Allows Facilities to look up an asset (room, door, desk, vehicle) to find the keys that operate it.

Project Process

How did we work on the project?

Planning	Analysis	Design	Implementation	Maintenance
Understanding and figuring out the problems	Analysing the Problem and Capital Data Requirements	Creating specifications for solutions	Building, testing and deploying solution	Monitoring and supporting application
Project Proposal	Data Requirement Table/ Conceptual Data Model Diagram	Logical Data Model Diagram	SQL codes/Application	

Structure of the Database System

What does it look like?

Key

Keyholder

Asset/Building

Conceptual Data Model

Data Requirement Table - Key

Entities and Attributes			
Entity	**Attribute**	**Props**	**Descripion**
Key	KeyStamp	RU	Code stamped on key
	Blank		Key blank needed to cut key
	Bitting	RU	Machine bitting to cut key
	Description		General info
	Hook	RU	Where made key is hung
	Made	D	Number made - generated by log
	Found	D	Number found - generated by log
	Returned	D	Number returned - generated by log
	Issued	D	Number issued - generated by log
	Destroyed	D	Number destroyed - generated by log
	Lost	D	Number lost - generated by log
	Available	D	Number on hand - Creat+Fnd+Rtrn-Issue-Dstr-Lost
KeyLog	Event ID	RU	System generated
	KeyStamp	R	Code stamped on key
	EventType	R	Made, found, returned, issued, etc
	event_date	R	Date
	Quantity	R	Number of keys
	KeyHolder	R	Keyholder assigned
	Authorized by	R	Facilities staff assigning key
Blank	Code	RU	Code stamped on blank key
KeyClass	Type	RU	Master, not master
Hook	ID	RU	cabinet & hook # (A1-400, B1-350, C1-120, D1-80)

- **Key: Key Specification**

- Key stamp(id), Status (made, found, destroyed, etc.), Brief Description

- **KeyLog: Key Registration Record**

- Event (Registration) id, date, key holder, etc.

- **Blank, Hook, KeyClass: Additional Information**

Conceptual Data Model

Data Requirement Table - Keyholder

Entities and Attributes			
Entity	**Attribute**	**Props**	**Descripion**
KeyHolder	Employee ID	RU	Org employee ID
	Employee Name	RC	Employee name
	Date Added	R	Date the user record was added
	Comments		General info
KeyHeld	Employee ID	RU	From KeyHolder table
	KeyStamp	RU	From key table

- **KeyHolder: The information who holds the key**

- Employee Id, and Date the user record was added, brief comments

- **KeyHeld: Connection between KeyHolder and Keys Table**

Conceptual Data Model

Data Requirement Table - Building

Entities and Attributes			
Entity	**Attribute**	**Props**	**Descripion**
Asset	Asset ID	RU	System generated
	Type	R	Door, desk, padlock, etc
	Asset	R	Friendly name
	Building	R	Building/ Property
	Room	R	Room number
	Location		Additional location info
	KeyStamp	R	Code stamped on key
	Rekeyed		Date item is rekeyed (rarely happens)
	Comments		General info
Building	ID	RU	System generated
	Name	RU	Friendly name
	MasterKey	R	Building master key
	Description		General info
	Address	RC	Building address
KeyAccess	KeyStamp	RU	From Key table
	Asset ID	RU	From Asset table
Asset Type	Category	RU	Alarm, locker, cabinet, door, padlock, desk, vehicle
MasterKey	KeyStamp	RU	From Key table (only for key class 'master'
	Building ID	RU	From building table
BuildingKey	KeyStamp	RU	From Key table
	Building Name	RU	From Building table

- **Building: The information of each building**

- Related master key, name, address, etc.

- **Asset: The information of each asset**

- Building, room, location, asset type, etc.

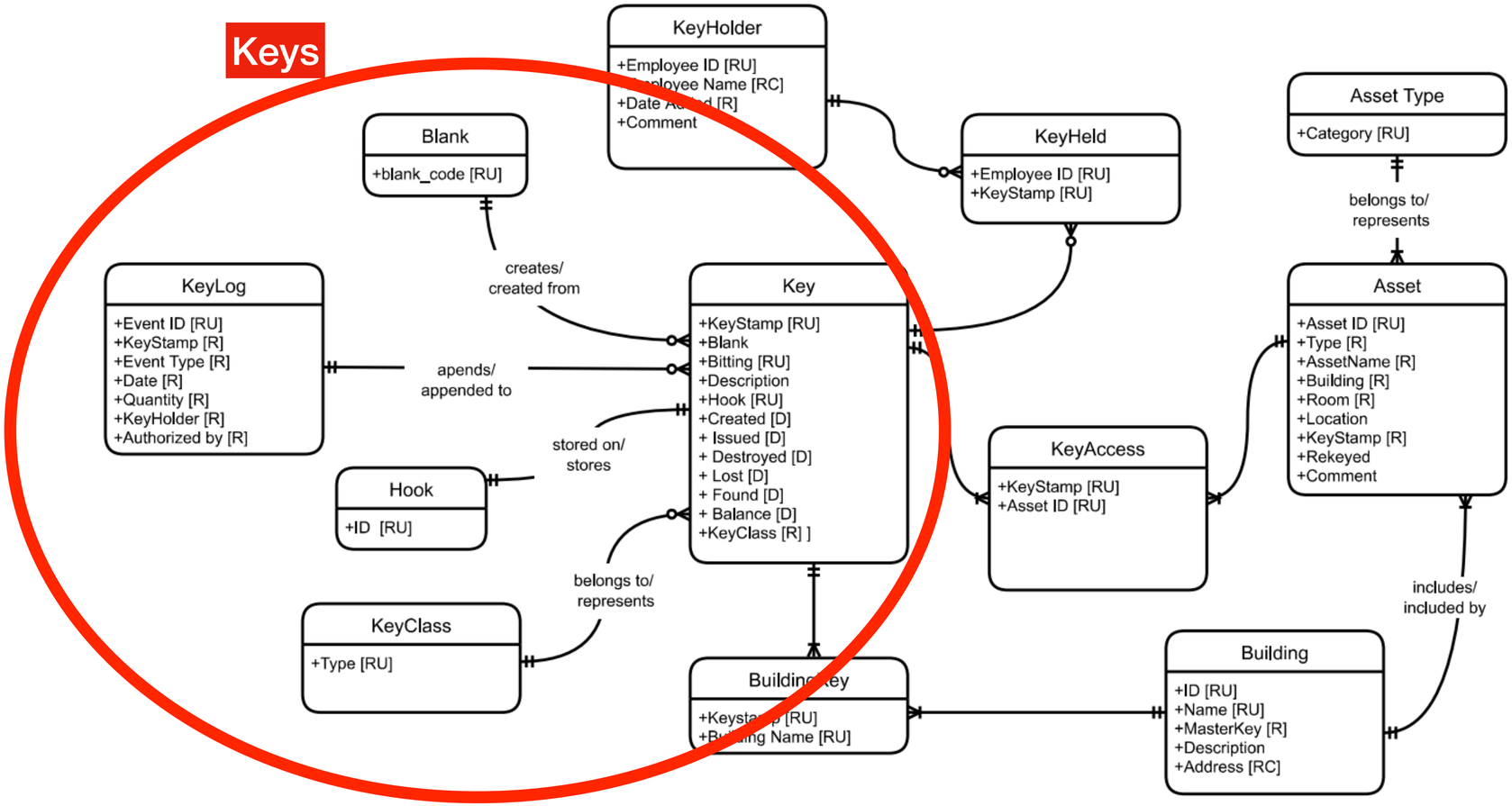
- **KeyAccess, BuildingKey: Connection between key and asset/building**

- **Asset Type: Additional Information of Assets**

- **Master Key: Managing for Only Master Key**

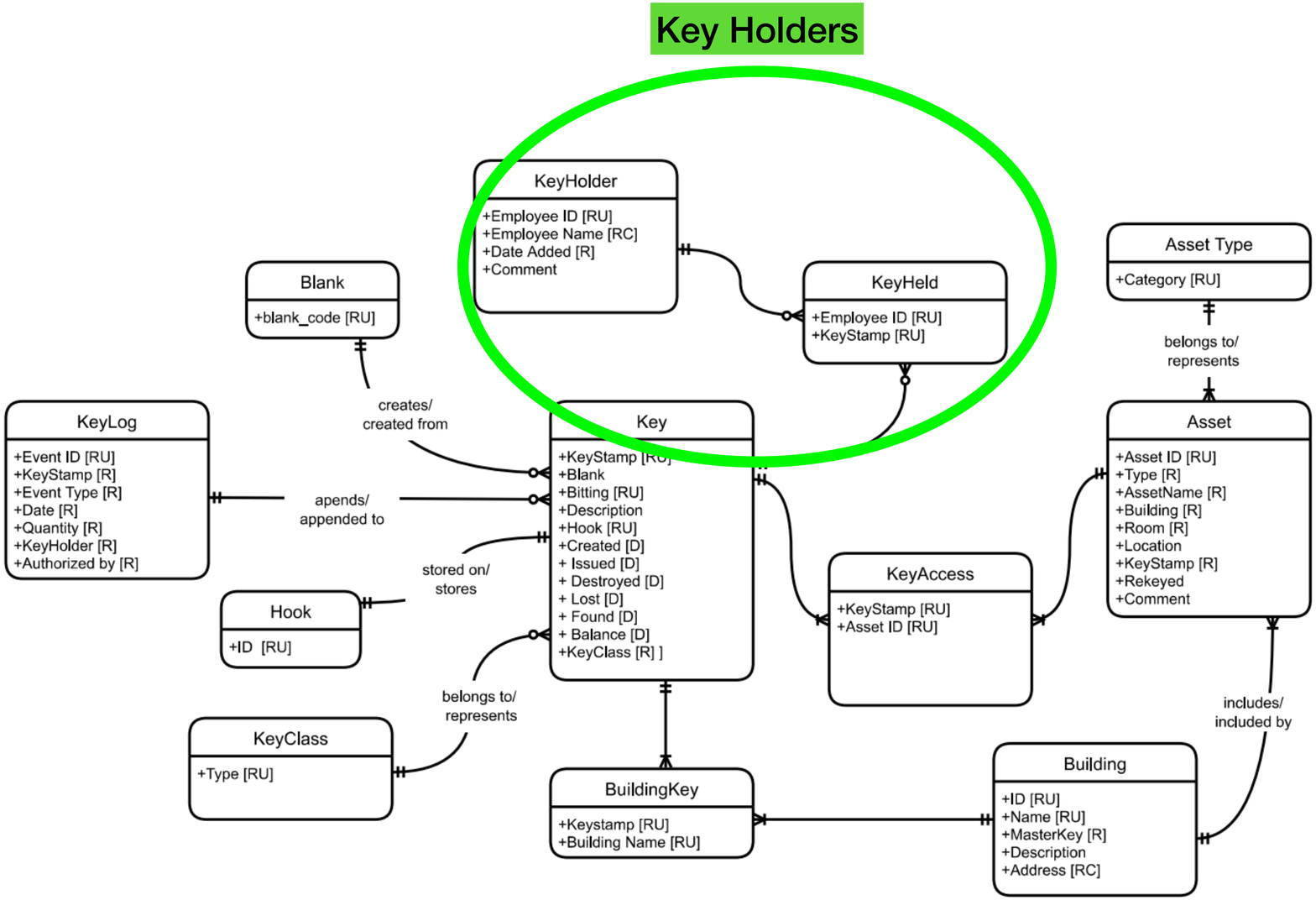
Conceptual Data Model

Diagram



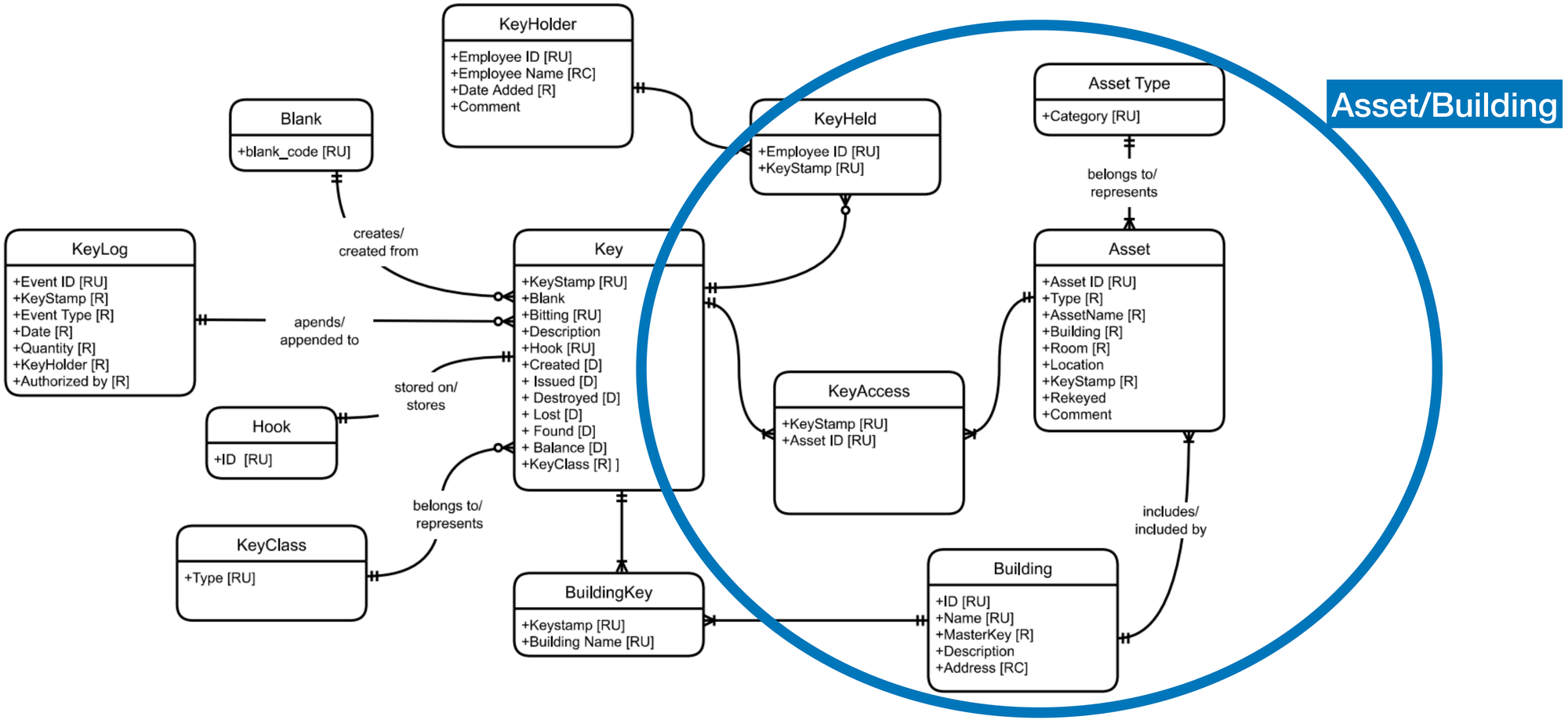
Conceptual Data Model

Diagram



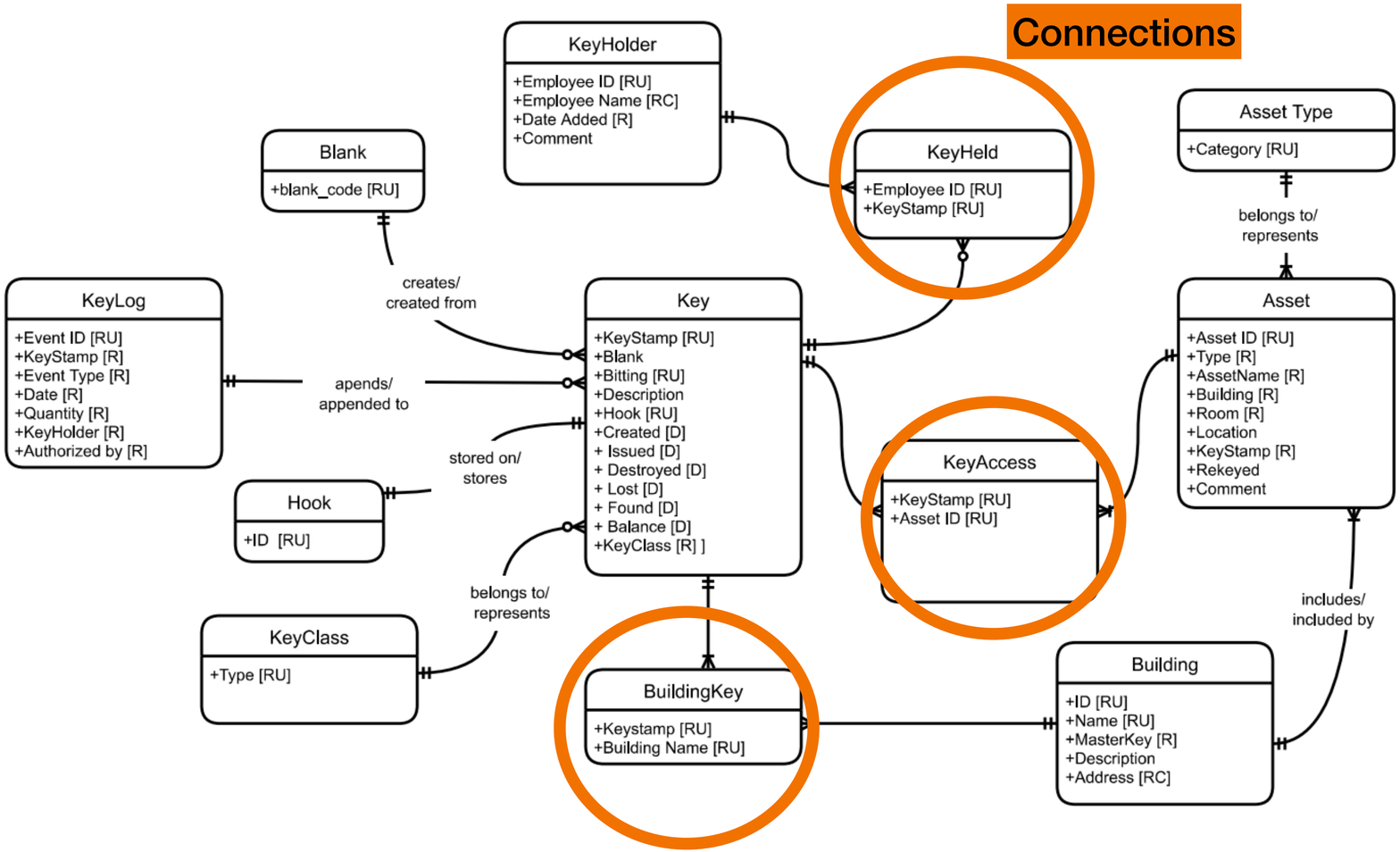
Conceptual Data Model

Diagram



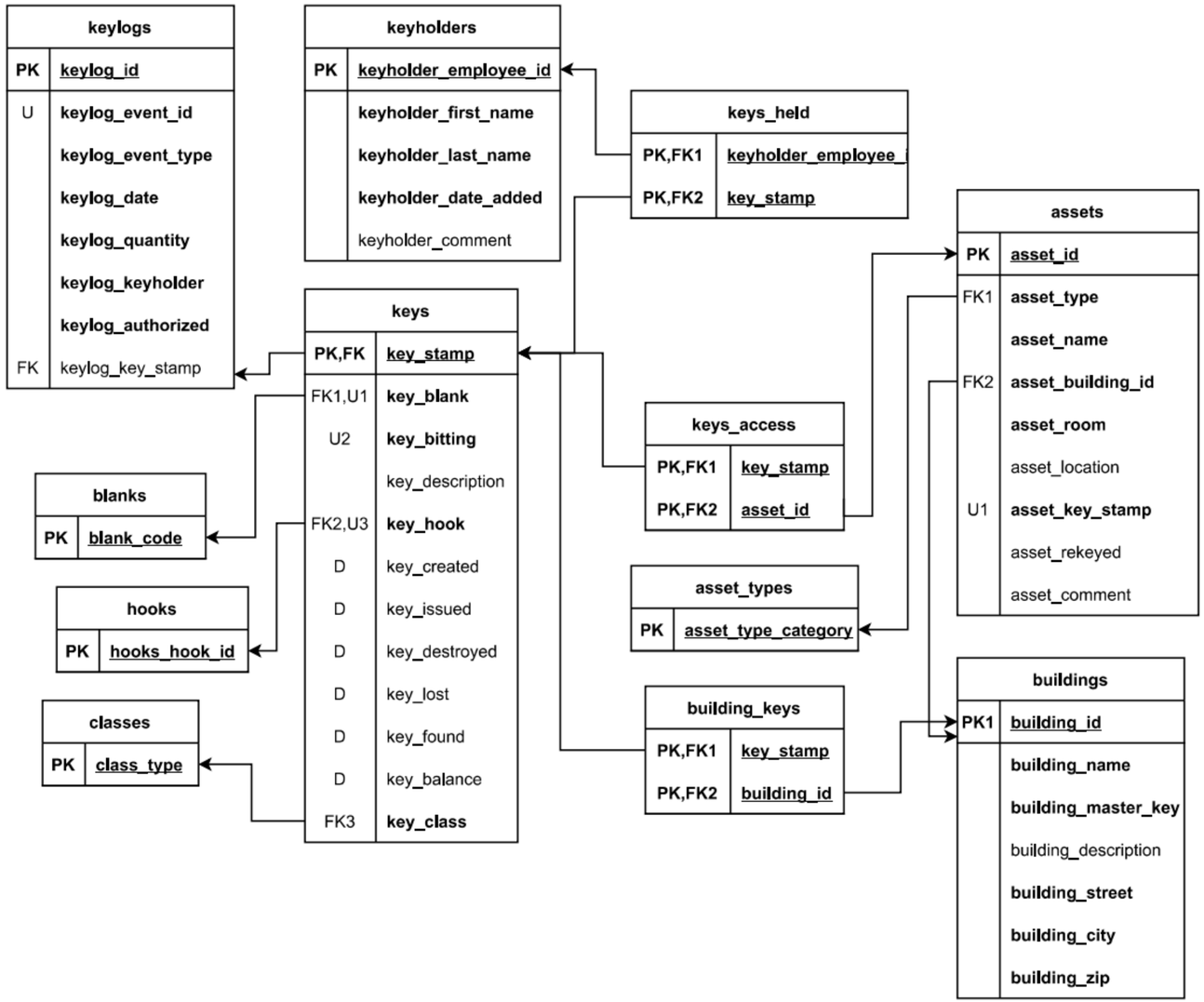
Conceptual Data Model

Diagram



Logical Data Model

Diagram



SQL Codes

Down/Up Script, Insert Value

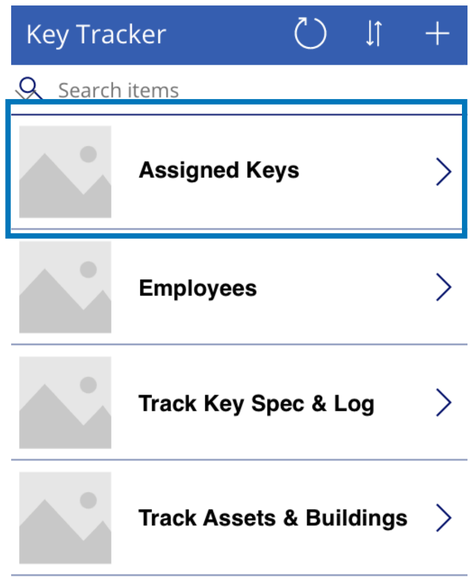
```sql
-- Down Script
-- (tables)

-- keylogs
IF EXISTS(SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE CONSTRAINT_NAME='PK_keylogs_keylog_id')
    ALTER TABLE keylogs DROP CONSTRAINT PK_keylogs_keylog_id

IF EXISTS(SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE CONSTRAINT_NAME='U_keylogs_keylog_event_id')
    ALTER TABLE keylogs DROP CONSTRAINT U_keylogs_keylog_event_id

IF EXISTS(SELECT * FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE CONSTRAINT_NAME='FK_keylogs_keylog_key_stamp')
    ALTER TABLE keylogs DROP CONSTRAINT FK_keylogs_keylog_key_stamp

DROP TABLE IF EXISTS keylogs
```

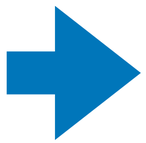
```sql
-- Up Script
CREATE DATABASE key_tracker

CREATE TABLE keylogs(
    -- attributes
    keylog_id int IDENTITY NOT NULL,
    keylog_event_id int NOT NULL,
    keylog_event_type varchar(20) NOT NULL,
    keylog_date date NOT NULL,
    keylog_quantity int NOT NULL,
    keylog_keyholder int NOT NULL,
    keylog_authorized_by int NOT NULL,
    keylog_key_stamp varchar(20) NOT NULL,
    -- primary key
    CONSTRAINT PK_keylogs_keylog_id PRIMARY KEY (keylog_id),
    -- unique
    CONSTRAINT U_keylogs_keylog_event_id UNIQUE (keylog_event_id),
    -- foreign key
    CONSTRAINT FK_keylogs_keylog_key_stamp FOREIGN KEY (keylog_key_stamp) REFERENCES keys(key_stamp)
)
```

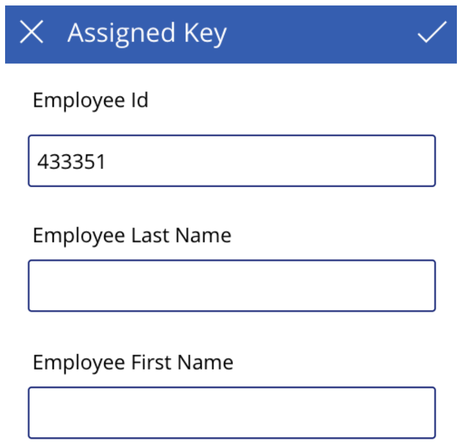
```sql
--keyholders table data insert
insert into keyholders (keyholder_employee_id, keyholder_first_name, keyholder_last_name, keyholder_date_added, keyholder_comment)
values
(616514,Elizabeth,Myers,"2022-12-04",'Signs out keys for IT Dept'),
(354654,Todd,Olden,"2022-12-04"," "),
(418651,Tyler,Reddo,"2022-12-04"," "),
(156678,Rick,Kent,"2022-12-04"," "),
(321562,Belinda,Sayer,"2022-12-04","CHRO"),
(746987,Suzie,Quaker,"2022-12-04"," "),
(439461,Frosty,Paws,"2022-12-04"," "),
(433351,Izzy,Girl,"2022-12-04"," "),
(596348,Pandora,DeGrey,"2022-12-04"," "),
(941683,Iggy,Quaker,"2022-12-04",'Limited access, loses keys');
```

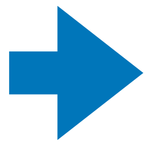
Applications

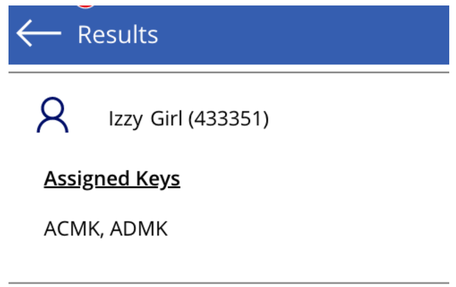
Function 1 - Viewing What Keys Are Assigned to an Employee.







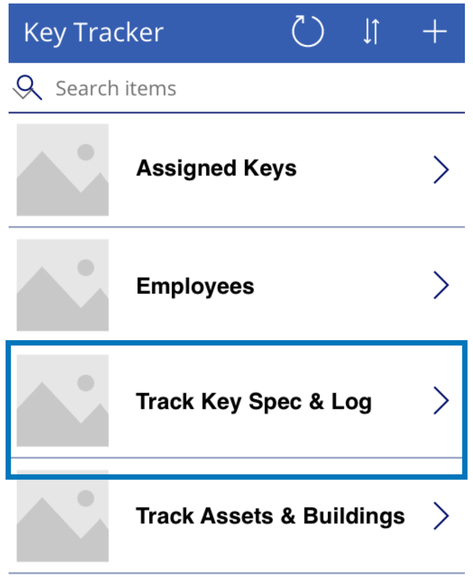




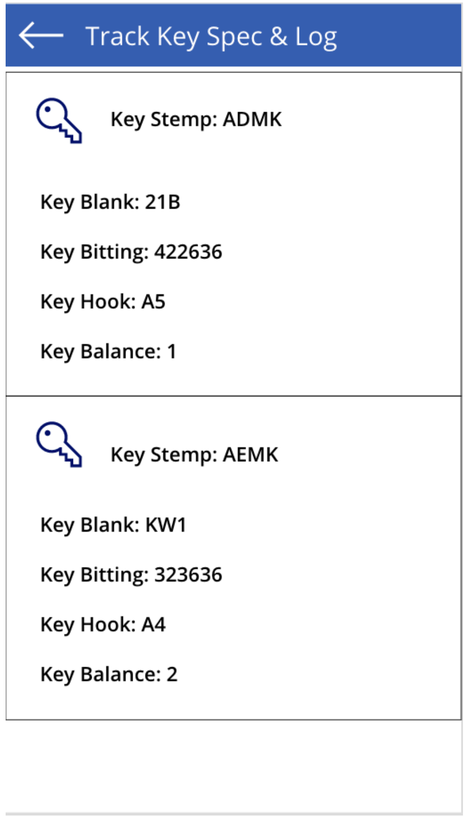

```
SELECT keys_held.keyholder_employee, keys.key_stamp
FROM keys_held
LEFT JOIN keys ON keys_held.key_stamp =keys.key_stamp
WHERE keys_held.keyholder_employee_id = [input] or
    key_held.keyholder_last_name LIKE '%[input]%'
        or key_held.keyholder_first_name '%[input]%'
ORDER BY keys_held.keyholder_employee;
```

Applications

Function 2 - Viewing All Employees That Are Assigned a Key

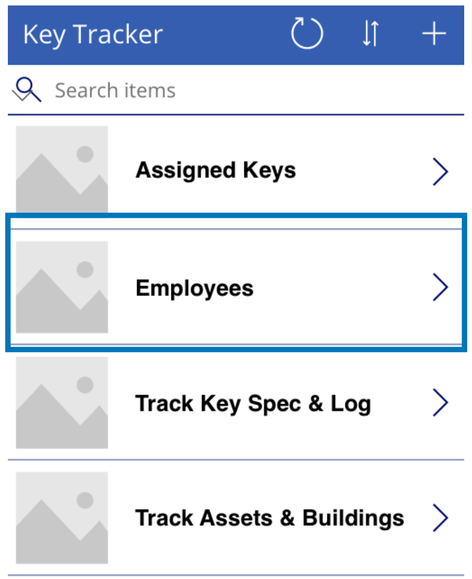


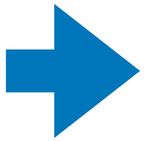

```
SELECT
key_stamp,key_bitting,key_hook,key_blank,
key_balance AS keys_on_hand
FROM keys;
```

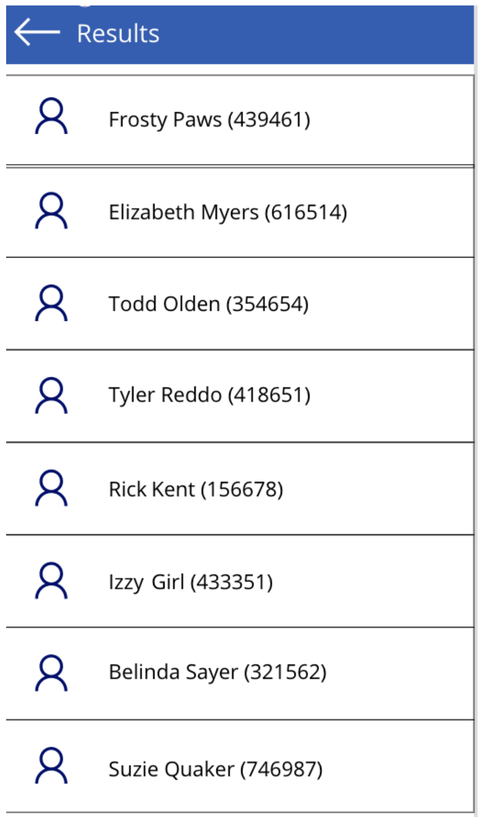


Applications

Function 3 - Track Key Data (Storage Hook, Bitting, Blank, Keys on Hand)





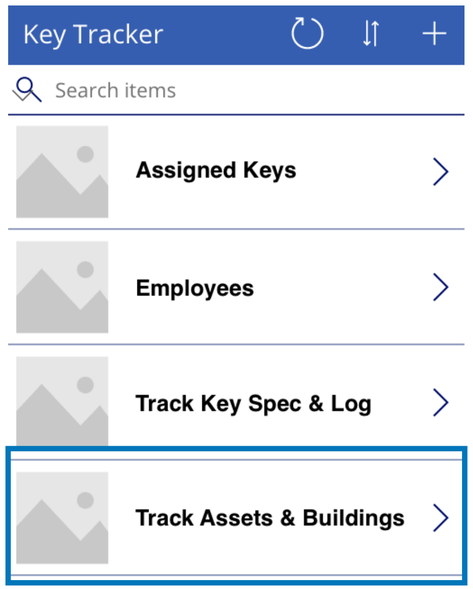


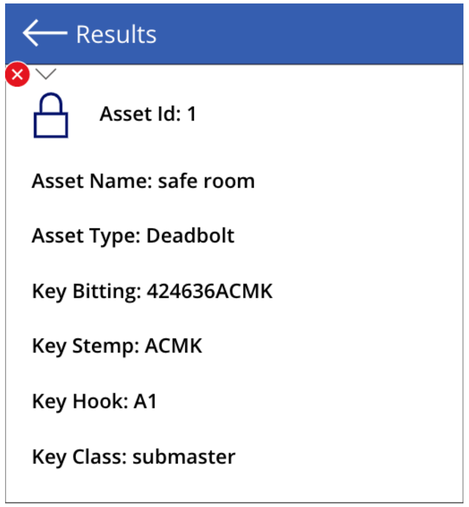
*** This function shows employees who has an assigned key.**

```
SELECT keyholders.keyholder_employee_id, keyholders.keyholder_first_name,
    keyholders.keyholder_last_name
FROM keyholders
RIGHT JOIN keys ON keyholders.keyholder_employee_id
    = keys_held.keyholder_employee
```

Applications

Function 4 - Track Assets & Building Data






```
SELECT assets.asset_id,assets.assest_type,assets.assest_name,
    keys.key_bitting,keys.key_stamp,keys.key_hook,keys.key_class
FROM assets
RIGHT JOIN keys_access ON keys_access.asset_id =assets.asset_id
RIGHT JOIN keys ON keys.keys_stamp = keys_access.key_stamp
WHERE assets.asset_id = [input];
```

Thank You