Team 37 – CSC2042 QA Project

Nathan Donaghy 40226743

[ndonaghy12@qub.ac.uk](mailto:ndonaghy12@qub.ac.uk)

David Mackenzie 40238376

[dmackenzie01@qub.ac.uk](mailto:dmackenzie01@qub.ac.uk)

Keelan Christopher Logan 40206821

[klogan03@qub.ac.uk](mailto:klogan03@qub.ac.uk)

Peter Sleith 40237264

[psleith01@qub.ac.uk](mailto:psleith01@qub.ac.uk)

Daniel White 40233631

[dwhite23@qub.ac.uk](mailto:dwhite23@qub.ac.uk)

**Entity Relationship Diagram**

A close up of a map

Description automatically generated

**List of Constraints and/or Assumptions in E-R Diagram Design**

A building can contain one or many apartments, but an apartment must be contained within one building.

From the assignment, it states that QA manages two disjoint categories of people, tenants and employees, therefore it is assumed than a person can be either an employee or a tenant but not both. There exists an “is a” relationship between these entities, i.e. an employee is an instance of a person, or a tenant is an instance of a person. Furthermore, employee can be either a technician, a manager or both. There also exists an “is a” relationship between these entities, i.e. a technician is an instance of an employee, and or a manager is an instance of an employee. Each person must also have their bank details recorded as employees are paid a salary and tenants must pay rent. The constraint in the ERD shows that each instance of a bank account must have one related person.

Each “Technician” can have one or more of the 3 outlined skills. Therefore, a “Skills” table was created to store the 3 skills and a separate TechnicianSkills table was created to store what skills each instance of a technician has. This was required for technicians with more than one skill in order to follow the relational database convention, otherwise it would be stored in a multi-valued attribute which is not allowed. Therefore, each Technician can have one or many TechnicianSkills, and each instance of TechnicianSkills must have one associated Technician. As well as this, each instance of TechnicianSkills must have one associated Skill and each Skill can have many associated TechnicianSkills.

In relation to “Lease”, a lease must be signed by one or more tenants, must be signed only one manager and must have only one related apartment. Another entity called “LeaseTenants” was created to allow the storage of multiple tenants under one lease agreement. This was necessary to ensure we were following relational database convention as multi-valued attributes are not allowed. A tenant can sign one or many leases as they may stay in different apartments over time, and the system will have both expired and current leases stored as required in the assignment. Additionally, a lease must have only one apartment, but each apartment can have one or many leases as the system will have both current and expired leases. Also, each lease must have one manager to approve the lease and each manager can approve one or many leases.

In relation to “Apartment” and “Manager”, by following the assignment document, it can be assumed than each apartment requires only one manager, therefore in the created ERD each manager manages one or many apartments and each apartment must have one manager.

In relation to “Office”, each manager is assigned to one office and each office can have one manager. Also, in the assignment it states that each office is located within an apartment owned by QA, hence in the ERD each office must be contained within one apartment and each apartment can have zero or many offices.

**SQL Querying – Query 1**

SELECT alt.LeaseID, alt.ApartmentNo, Building.BuildingNameorNum, Building.Street, Building.PostCode, (alt.NumBedrooms-alt.NumTenants) AS 'NumFreeBedrooms'

FROM (SELECT Apartment.ApartmentNo, Apartment.NumBedrooms, Apartment.BuildingID, lt.LeaseID, lt.NumTenants

FROM Apartment

INNER JOIN (SELECT LeaseTenants.LeaseID, LeaseTenants.TenantID, Lease.ApartmentID, COUNT(\*) AS 'NumTenants'

FROM LeaseTenants

INNER JOIN Lease ON LeaseTenants.LeaseID = Lease.LeaseID WHERE (SUBDATE(CURRENT\_DATE, INTERVAL Lease.Duration MONTH) < Lease.StartDate) GROUP BY LeaseTenants.LeaseID)lt ON lt.ApartmentID = Apartment.ApartmentID)alt INNER JOIN Building ON alt.BuildingID = Building.BuildingID WHERE (alt.NumBedrooms - alt.NumTenants >0)

QA wants to know which of its leases and apartments are still able to accommodate tenants and how many it can accommodate.

It finds current non-expired leases for apartments where there are more bedrooms than tenants living in the apartment.

It then displays the leaseID, ApartmentNo, Address and the Number of free bedrooms.