Project Team 3

Database Topic: Property Management System

Team Members: Chetna Khanna, Jay Chaudhari, Phani Sai Kamal Lingam, Roopa Uma Shiv

Database Specification: Purpose, Business problems addressed, Business rules

Database Purpose: The main purpose of the database is to maintain all the data related to residential properties and their residents, making the process of leasing hassle-free for the management and simple for the tenants. The database will also store information related to rental services and maintenance, so that, both the managers and the residents can schedule tasks and keep track of the progress.

Business Problems Addressed:

- Enables the management to keep track of all the resident information.
- Allows the management to track whether rent payments and all the other charges have been made by the residents on time and helps to identify the defaulters.
- Allows the staff members to keep track of the maintenance requests raised by the residents and monitor the status of those requests.
- Provides information relating to the resident response and participation to the community events hosted and gives more insight into what events are generally preferred by the larger audience.
- Gives insights into the current leasing demand trend and enables the management to strengthen the apartment promotions accordingly.
- Enables the management to keep track of the availability and correct usage of parking lots.

Business Rules:

- A block can have many apartments.
- An apartment can have one or more residents, but the residents of each apartment will have only one unique ResidentID.
- An apartment can have more than one or more pet and the rent for each pet should be paid.
- A resident can be allotted more than one parking lot.
- An apartment portfolio can have one or more similar type of apartments.
- Apartment promotions include the promotional offers that many apartments have.
- A staff member can sign lease of one or many future residents.

- A resident can have one or more emergency contacts.
- A resident can have one or more lease over the duration of his/her stay in the apartment.
- A resident can take part in one or more community events.
- A bill can be paid by making one or more payment transactions.
- A staff member can be assigned with multiple maintenance requests.
- A resident can raise one or more maintenance requests during his/her stay.
- A staff member can handle multiple community events.
- An apartment can have one or more utilities and each with separate utility charges.
- An apartment can have one or more amenities and each with separate charges.
- Only one bill will be generated for insurance for a particular term length chosen by the resident.

Design Requirements:

- Used crow's foot notation.
- Drawn dashed lines between any two entities in order to show the non-identifying relationship.
- Specified which table should be on the one side in the relationship by placing a 1 next to the table where the line starts.
- Specified which table should be on the many side in the relationship by placing a crow's foot symbol next to the table where the line ends.

Design Decisions:

Entity Name	Why entity included?	How an entity is related to other entities?
Blocks	In general, a property comprises of many apartments, so for the purpose of identification, the apartments are grouped into blocks with a unique id (BlockID) and a name to represent each block.	The Blocks entity's primary key BlockID relates to the Apartments entity. These two entities have one-to-many relationships from Blocks entity to Apartments entity, as one block can comprise of multiple apartments and each apartment can belong to only one block.
Apartments	The management needs to know the availability status of each apartment	As an apartment can have many utilities, amenities, residents and pets, the

	for leasing purpose. The Apartment entity is created to help management identify the block to which a particular apartment belongs to and the spatial and rental details of that apartment.	primary key ApartmentID of the Apartment entity relates to Utilities, Pets and Residents entities in one-to-many relationships. Similarly, an apartment can have multiple lease records with several insurance details associated to an apartment over that period of time. So, there is a one-to-many relationship between Apartments entity and Insurance entity. A single block contains multiple apartments, but an apartment can be present only in a single block. Similarly, the portfolio collections maintained by the management contains details of all the apartments, but a single apartment can have only one portfolio. Therefore, Apartments entity relates to both Block and ApartmentPortfolio entities in many-to-one relationship.
ApartmentAddress	Every Apartment have its own unique Postal Address for legal purposes. Without this it would impossible to identify apartment in global scale.	Each Apartment has a unique mandatory Apartment Address. These are associated with a One to One relationship with ApartmentID as foreign key in ApartmentAddress Entity.
ApartmentMaintenanceRecords	The apartments undergo several renovations and maintenance every year. It is useful to keep track of these servicing logs so that we can identify the	The ApartmentMaintenanceRecords Entity is associated with the Apartments Entity to keep a history of all maintenance undergone

	problems and the causes.	with one to many relationships with ApartmentID as foreign key. This is optional which means in case of new Apartments there can be no maintenance records.
Amenities	The Amenities entity includes the desirable features and useful facilities available and accessible by the residents of an Apartment. This includes facilities like gym, swimming pool, club house, party rooms, elevator, sun deck, cyber hub etc. Some of these are based free of charge and some are charged per use.	The Amenities entity is tied with the Apartments Entity with one to many relationships as there can be multiple amenities accessible by the residents of an apartment. The ApartmentID is used as a foreign key in the Amenities table to show accessible facilities to the residents.
Utilities	The Utilities entity consists of the services being provided to the residents of an apartment. Utilities include services like Electricity, sewerage, gas, garbage disposal, water, telephone, internet etc.	Similar to the amenities entity the Utilities entity is also consists of the ApartmentID attribute to entitle what all services are being provided to a particular apartment. This is made possible by using one to many relationships making multiple utilities serviceable to a single apartment.
Appliances	All the electronic devices or equipment which are included in the apartment. These include refrigerator, oven, microwave, dishwasher, laundry, toaster etc.	Each apartment is provided with different set of appliances based on the type of apartment or size of apartment. So, each appliance has its own unique identifier ApplianceID along with the ApartmentID as a foreign key to know what all devices are there in the apartment. A one to many

		relationships is created between the Apartments table and Appliances table denoting that one apartment can have multiple appliances.
Insurance	It is mandatory for an apartment to be included under a renter's insurance scheme for every lease period. So that in case of any mishap the insurer will be covering up the losses.	With the change in ownership of an apartment after the end of a lease term, the insurance scheme associated with that apartment might get changed to a different one. Therefore, the primary key InsuranceID relates to Apartment entity in a many-to-one relationship. Similarly, only one bill will be generated for one InsuranceID for a particular term length chosen by the resident (it can be on a monthly, quarterly, bi-yearly or yearly basis).So, there exists a one-to-one relationship between Insurance and Billing entities.
Billing	The Billing entity aggregates all the relevant usage charges of an apartment such as apartment rent, utilities charges, parking fees etc. This also store the information crucial to check the due dates, total amount etc.	Entities like Utilities, Insurance, Pets etc., are linked to the Billing Entity so that it can account for all the necessary charges based upon the usage by the residents of the apartment or the apartment in whole. These relationships are established by one to one and one to many relationships meaning every different utility has different usage charges.
Payments	The Payments entity holds	The Payments entity is

	the records of monetary transactions which are made in order to pay the billed amount. This also included the payment method or mode, status, date etc.	linked with the Billing entity by a one to many relationships, meaning that for a single Billing there can be multiple payments made in portions or due to transaction failures. But it is not possible that one payment record accounts for multiple billing statements.
ApartmentPortfolio	Whenever a person visits a property to see an apartment, the property manager shows him/her an apartment portfolio. The portfolio contains details of the apartment alongwith its availability status and date. Each apartment portfolio has a unique ApartmentPortfolioID.	The ApartmentPortfolio entity with primary key ApartmentPortfolioID is associated with Apartments entity by one-to-many relationship as an apartment portfolio can have more than one apartment of the similar type. Also, the ApartmentPromotions entity is linked to ApartmentPortfolio entity by one-to-many relationship as any particular promotion can cover many apartments which are listed in the apartment portfolio.
ApartmentPromotions	The ApartmentPromotions entity comes into play during a particular time period when there are some ongoing promotional events like waiving of the security deposit or first month rental charges, agent referral discount etc., which are used to attract new residents to the community.	The ApartmentPortfolio consists of all the Apartments available for leasing purposes. The promotions are generally offered during this period of leasing. The ApartmentPromotionID is included in the ApartmentPortfolio table making it easy to check the promotional offer on the available apartments. Both entities are related by a

		Many to One relationship such that many Apartments can have a single promotion during a time period but cannot have multiple promotional offers.
Residents	One of the most crucial details in the property management database is the resident information. The Resident entity is created to include the personal information of each resident and a unique id (ResidentID) to commonly represent the resident and co-residents of a particular apartment.	The resident information is used by many other entities in the database. The Resident entity's primary key ResidentID relates to Lease, ParkingLots, Billing and MaintenanceRequests in one-to-many relationships. There is a many-to-one relationship between Residents entity and Apartments entity, as an apartment can have one or more residents. As there is a many-to-many relationships present between Residents entity and CommunityEvents entity, an associative entity named ResidentCommunityEvents is created to satisfy the normalization rules.
ResidentsEmergencyContacts	In case of occurrence of any incident related to the residents, the management needs to have a point-of-contact for each resident in the property.	The ResidentEmergencyContacts contains the name, address and contact information of the point-of-contact of the residents. The primary key ResidentEmergencyContactID relates to Resident entity in many-to-one relationship, as each resident can provide one or more emergency contact details to the management.

ResidentEmploymentRecords	For understanding the economic background of the resident, it is necessary to keep a record of his current occupation and it also helps to contact the resident in case of emergency where he is unavailable.	The ResidentEmploymentRecords are associated with Residents Entity with one to one relationship and is also optional that means a student won't have a employment so it can be null. The ResidentID is denoted as a foreign key here.
ResidentCompanyAddress	The company address is needed for the ResidentEmploymentRecords. In order to achieve third normal form, we need a separate entity to store the postal address of the Company.	Each Company will have a Postal Address. So, ResidentCompanyAddress is associated with ResidentEmploymentRecords with a mandatory one to one relationship. This is acheived by denoting ResidentEmploymentRecordID as a foreign key.
Pets	People love pets and some people live with those pets. Pets entity accommodates the cases of such people and helps in keeping a record of the pets associated with the residents and charging them based on the number of pets, pet type or pet size etc.,	The Residents Entity and Pets Entity are associated with a one to many relationships. This allows the residents of an apartment to have multiple pets. Hence, the ResidentID is denoted as a foreign key in the Pets Entity having the association between the pet and owner of pet clear. This also optional enitity which means there can be no pets as well.
Lease	Apartments can in general be in demand during the peak seasons and in order to facilitate bookings by future residents, the management needs to know the lease duration of each and every apartment. Management can collect	The primary key LeaseID of the Lease entity relates to Residents and StaffMembers entity in many-to-one relationships. This is because, one staff member can be incharge of lease details of multiple residents and the lease of a

	information such as whether the current residents have a plan to extend their lease term or not, by sending out lease extension notifications before the end of each lease term to corresponding residents.	particular apartment can be handled by only one staff member. Also, a resident can take lease in the same apartment or another apartment within the same property multiple times during a span of several years, but at a given time a lease can only be linked to one ResidentID.
ParkingLots	ParkingLots entity is included to keep track of the parking spaces available or allocated to the residents within the block and also in the community. The entity also carries information such as parking fees, type of parking spaces like covered ones, open ones etc.	The Residents entity is connected to the ParkingLots entity through 1 to Many relationships. Which means that a Single Resident can have multiple parking lots assigned but a single Parking Lot cannot be assigned to multiple Residents as it violates the business rule. The ResidentID is created as a Foreign Key in the ParkingLots entity which is connected to the ResidentID which is a Primary Key in the Residents Table.
StaffMembers	For any system to work smoothly, we need staff. In order to make our property management system work smoothly we have staff members with different designations, who play different roles towards keeping the property well maintained. These staff members are identified by their unique StaffMemberID.	Each staff member has a specific role in our system. A few works towards maintaining the society while there are a few who works to make the stay of the residents happy by organising various community events. There are also a few who works for the finance and management department and helps future residents in finalising the apartment and sign their leases. The StaffMember entity with

		primary key (StaffMemberID) relates to the Lease, MaintenanceRequests, StaffMemberCommunityEvents entities by one-to-many relationships as a staff member can sign one or many leases, can handle one or many maintenance requests and can organise one or many community events.
StaffMembersEmergencyContacts	In case of occurrence of any incident related to the staff members, the management needs to have a point-of-contact for each staff member in the property.	The StaffMembersEmergencyContacts contains the name, address and contact information of the point-of-contact of the staff members. The primary key StaffMembersEmergencyContactID relates to StaffMembers entity in many-to-one relationship, as each staff member can provide one or more emergency contact details to the management.
MaintenanceRequests	During the course of stay in an apartment, residents may encounter some (or many) issues related to the utilities, amenities, apartment interior, exterior etc. All these issues are raised in the form of maintenance requests, each identified by the unique MaintenanceID. These requests are then handled by the staff members in order to make the duration of stay of residents pleasant.	The maintenance requests are stored using the MaintenanceID, which is the primary key of this entity. This unique ID helps the property managers to track the maintenance record. The requests are raised by the residents and assigned to the staff members, who works in the maintenance department. So, the MaintenanceRequests entity has a relationship with both Residents and StaffMembers entity. As a

		resident can raise one or many requests and a staff member can address one or many requests, the relationships are made one-to-many towards the MaintenanceRequests side.
CommunityEvents	Community events can help build cordial relationships between residents and strengthen the community. For this purpose, every residential property organizes several events throughout the year. We have included CommunityEvents entity with the same purpose and assigned a unique CommunityEventsID to each event.	A community event is generally organised by staff members and many residents take part in it. The CommunityEvents entity with primary key CommunityEventsID is associated with residents and staff members by many-to-many relationships as one or many events can be organized by one or many staff members and one or many residents can be a part of it. In order to avoid many-to-many relationships, we have included the associative entities named ResidentCommunityEvents and StaffMembersCommunityEvents.
StaffMembersCommunityEvents	In order to organize community events, an active participation of staff members is required. But, since staff members and community events have many-to-many relationships, we created an associative entity named StaffMembersCommunityEvents having unique StaffMembersCommunityEventsID.	The associative entity StaffMembersCommunityEvents with primary key StaffMembersCommunityEventsID is linked to StaffMembers and CommunityEvents entities by many-to-one relationships. The associative entity has been added in order to satisfy the normalization rules.

ResidentCommunityEvents	A community event can have a lot of residents as participants and a resident can take part in many community events. So, the relationship between Residents and CommunityEvents entity is many-to-many and in order to avoid this, we have created an associative entity named ResidentCommunityEvents.	In order to satisfy the normalization rules, an associative entity named ResidentCommunityEvents has been created. The entity ResidentCommunityEvents with primary key ResidentCommunityEvents ID is linked to Residents and CommunityEvents entities by many-to-one relationships.