BUAN 6320 DATABASE FOUNDATIONS FOR BUSINESS ANALYTICS

PRESENTED TO VIVEK MISHRA:

LAKE RIDGE APARTMENTS

Group Name: LUDO

BY GROUP MEMBERS:

Ravikiran Ramchandra Pise
Sangram Tushar Mohite Patil
Chinmay Mukesh Sinari
Tejas Kadu

TABLE OF CONTENTS

Problem Description

Problem Statement

Lake Ridge Apartments is an up and coming apartment complex in the suburbs of Dallas and has been facing a steady influx of investment in the community to expand its resources and offerings to the public. Through this project, Lake Ridge Apartments would be able to effectively manage different aspects of the complex and the services offered to tenants while managing the employee hierarchy as well.

Organization Description

Company Overview

Lake Ridge Apartments is a housing complex with state of the art amenities and services at affordable rates for the people and hosts a variety of housing options. Following are the different aspects of the organization

Employees
Lake Ridge employs 16 people and has a defined hierarchy and segregation between the employees based on the nature of the work ranging from the General Manager to Community Officers

Apartments and their description
The apartments are of 4 types and in 10 categories of areas. The largest being a 2B2B with an area of 1500sq ft and the smallest being a 1B1B with 650 Sq ft.

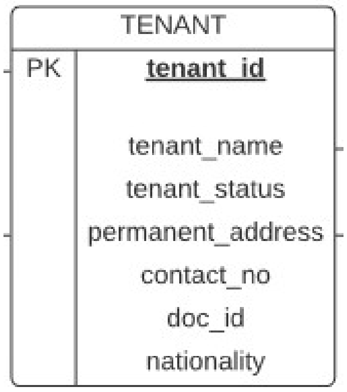
Tenants
The complex hosts 14 tenants from different nationalities and places and offers pre-booking of the apartments allowing in future tenant-lender relationship

Services
In addition to these two offerings, the complex also has Services like Pools, Community center, Game rooms, etc. to offer to the tenants. There is a check which requires the employee of the organization to sign-off for usage

Scope of Database

Tenant Table

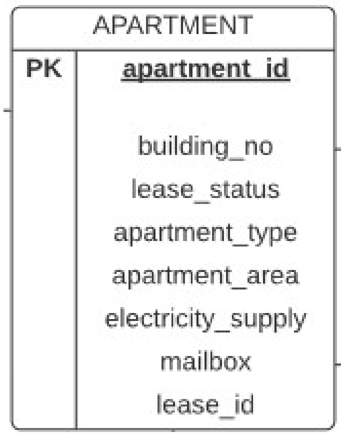
The Tenant table as the name suggests hosts the information about the tenants in the complex. Following is the table structure. *'tenant_id'* is the primary key of the table with tenant statuses being *'ACTIVE', 'INACTIVE', and 'EXPIRED'.*



TENANT	
PK	tenant_id
	tenant_name
	tenant_status
	permanent_address
	contact_no
	doc_id
	nationality

Apartment *Table*

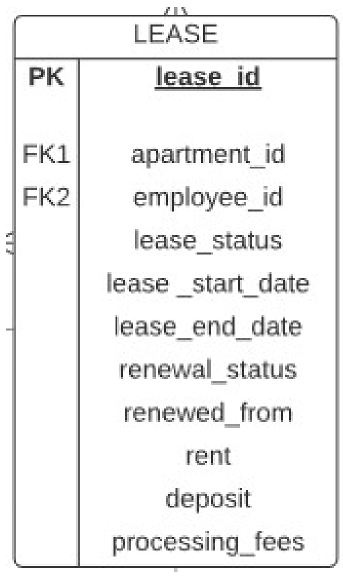
The ***APARTMENT*** table is the most important table of the system hosting and storing the information about the apartments in the complex. The apartment table hosts the lease status and just like the status in the tenant table, has statuses *'ACTIVE', 'INACTIVE', and 'EXPIRED' status*



APARTMENT	
PK	apartment_id
	building_no
	lease_status
	apartment_type
	apartment_area
	electricity_supply
	mailbox
	lease_id

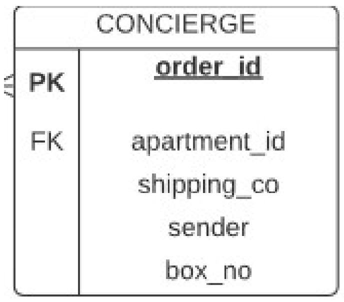
Lease *Table*

The LEASE table has the relevant information about the leases for the apartments. The **key business logic to note is that the leases are signed for the apartments and not the tenants**. Hence the direct link is established with the apartment table with ***lease_id*** being the primary key of the table and *apartment_id* being the foreign key. A lease has to be signed and processed by an employee hence the link to the EMPLOYEE table



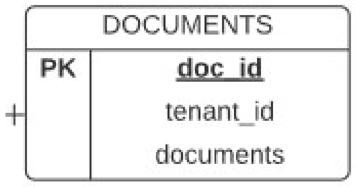
LEASE	
PK	lease_id
FK1	apartment_id
FK2	employee_id
	lease_status
	lease _start_date
	lease_end_date
	renewal_status
	renewed_from
	rent
	deposit
	processing_fees

Concierge *Table*
Information about packages and deliveries are stored in the Concierge table where it is related to an apartment and the fields corresponding to it are – *box_no, sender, shipping_co* – the primary key being ***order_id***



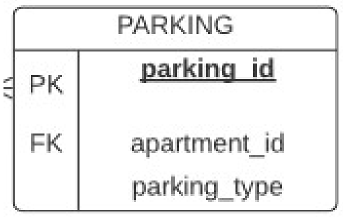
CONCIERGE	
PK	order_id
FK	apartment_id
	shipping_co
	sender
	box_no

Documents *Table*
The DOCUMENTS table holds the information about the documents to each tenant. Each tenant has a document id and the primary key of the table is *doc_id*.



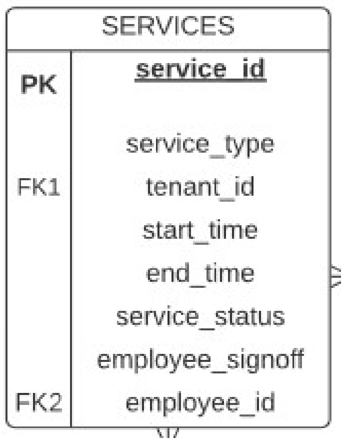
DOCUMENTS	
PK	doc_id
	tenant_id
	documents

Parking *Table*
Information about the parking services offered to the apartment is stored in the *PARKING* table. Please note that the parking is allotted for an apartment and not for a tenant. Hence the link is to the Apartment table.



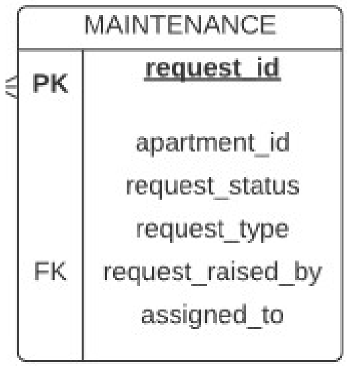
PARKING	
PK	parking_id
FK	apartment_id
	parking_type

Services *Table*
The *SERVICES* table holds the services availed by the tenants. The service requests have to be signed off by the employees of the complex and employing the check on the service consumption. The *service_id* is the primary key of the table with 2 foreign keys to the *TENANT* and *EMPLOYEE* tables. The statuses for the requests are Received, In-progress, Done



SERVICES	
PK	**service_id**
	service_type
FK1	tenant_id
	start_time
	end_time
	service_status
	employee_signoff
FK2	employee_id

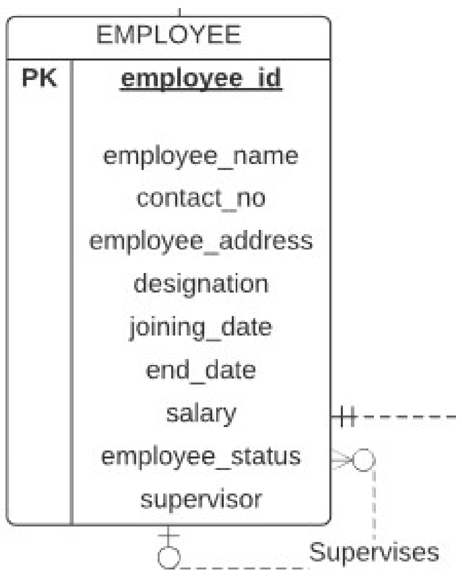
Maintenance *Table*
Any maintenance requirements submitted by the tenants are logged in the *MAINTENANCE* table. The statuses for the requests are Received, In-progress, Done. The primary key of the table is *request_id* with a link to a tenant because the maintenance requests are raised by the tenants



MAINTENANCE	
PK	**request_id**
	apartment_id
	request_status
	request_type
FK	request_raised_by
	assigned_to

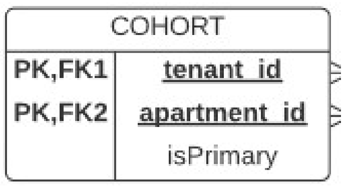
<u>Employee *Table*</u>

EMPLOYEE table is a crucial table in the system hosting information about the employees, their duties, contact numbers, supervisors, and having links to other crucial tables – *LEASE, SERVICES, MAINTENANCE*. ***employee_id** is the primary key*

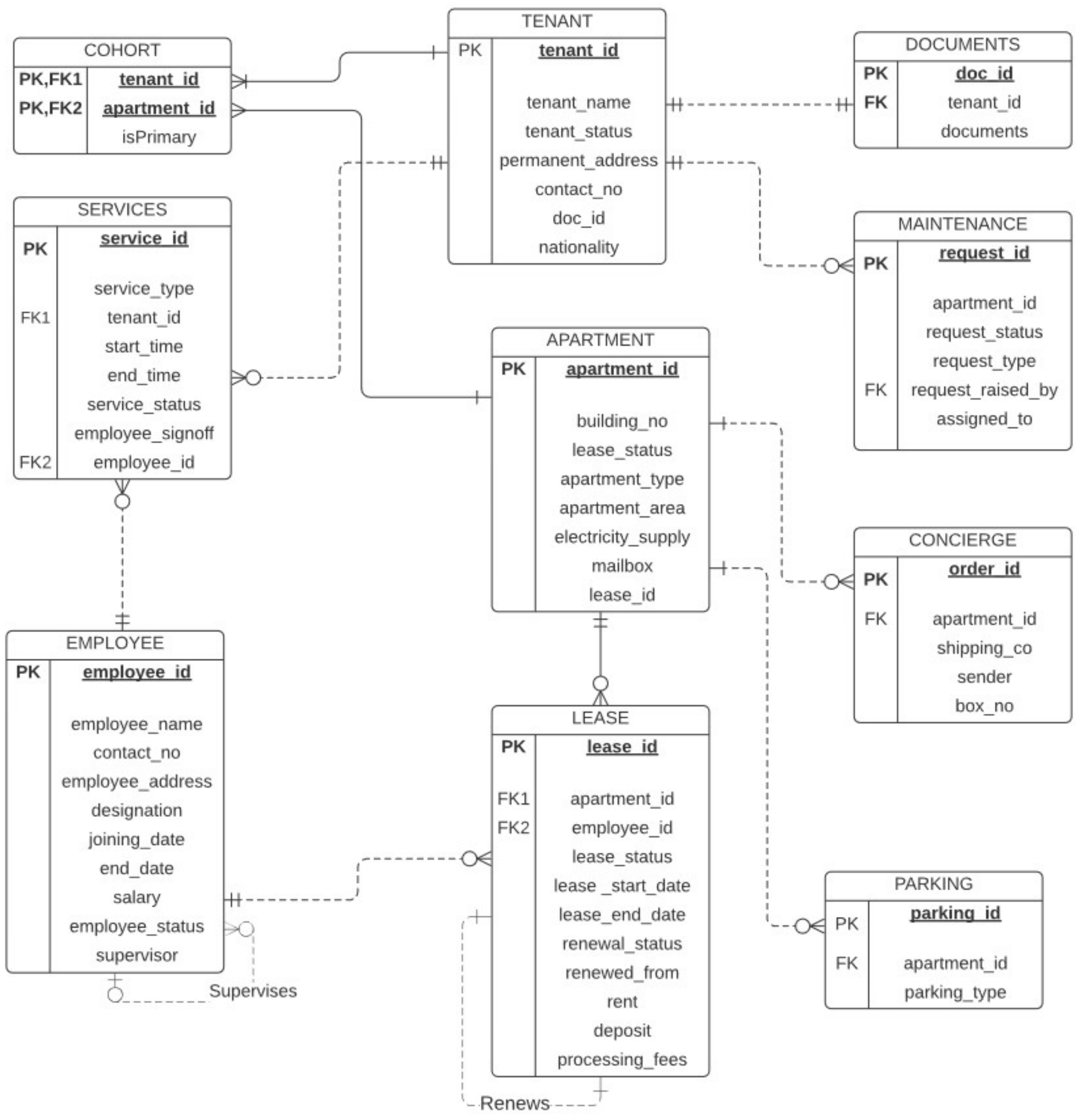


<u>COHORT *Table*</u>

COHORT table is a crucial table in the system linking the TENANT and APARTMENT tables. It has 3 columns apartment_id, tenant_id, and isprimary. The *apartment_id* and *tenant_id* keys are grouped together to form a grouped primary key



Entity Relationship Diagram



Relational Database Schema

APARTMENT Table

```
create          table     APARTMENT(
  apartment_id        INT     NOT    NULL ,
  building_no         INT     NOT    NULL,
  lease_status        VARCHAR(100)       NOT    NULL,
  apartment_type      VARCHAR(100)       NOT    NULL,
  apartment_area      INT     NOT     NULL,
  electricity_supply  VARCHAR(200)       NOT    NULL,
  mailbox            INT      NOT     NULL ,
  lease_id           INT      NOT     NULL,

  PRIMARY KEY (apartment_id)
  );
```

TENANT Table

```
create          table     TENANT (
  tenant_id   INT      NOT   NULL ,
  tenant_name         VARCHAR(100)       NOT    NULL,
  tenant_status       VARCHAR(100)       NOT    NULL,
  permanent_address VARCHAR(200)        NOT    NULL,
  contact_no bigint   NOT    NULL ,
  doc_id              INT     NOT   NULL,
  nationality         VARCHAR (200) NOT NULL,

  PRIMARY KEY (tenant_id)
  );
```

COHORT Table

```
create          table          COHORT (
      tenant_id       INT NOT NULL,
      apartment_id    INT NOT NULL,
      isPrimary       BIT NOT NULL,

      constraint      PK_COHORT          PRIMARY KEY       (tenant_id,apartment_id),
      constraint      FK_COHORT_TENANT FOREIGN KEY  (tenant_id)      references
      TENANT(tenant_id),
      constraint      FK_COHORT_APARTMENT FOREIGN KEY (apartment_id)
      references APARTMENT(apartment_id)
);
```

<u>CONCIERGE Table</u>

```
create          table     CONCIERGE (
  order_id     INT       NOT    NULL ,
  apartment_id INT   NOT    NULL,
  shipping_co VARCHAR(200)      NOT    NULL,
  sender        VARCHAR(200)      NOT    NULL,
  box_no        INT                       NOT    NULL,

  PRIMARY KEY            (order_id),
  CONSTRIANT     FK_CONCIERGE FOREIGN KEY (apartment_id) REFERENCES
      APARTMENT(apartment_id)
 );
```

<u>DOCUMENTS Table</u>

```
create          table        DOCUMENTS (
                tenant_id   INT NOT NULL,
                doc_id       INT NOT NULL,
                documents NVARCHAR(MAX) NULL
                PRIMARY KEY (doc_id),
                constraint FK_DOCUMENTS FOREIGN KEY (tenant_id) references
                TENANT(tenant_id)
);
```

<u>EMPLOYEE Table</u>

```
create table EMPLOYEE (
      employee_id            INT                  NOT          NULL,
      employee_name         VARCHAR(200) NOT          NULL,
      contact_no              BIGINT             NOT          NULL,
      employee_address     VARCHAR(200) NOT          NULL,
      designation             VARCHAR(200),
      joining_date            DATE                 NOT          NULL,
      end_date                 DATE,
      salary                     INT                  NOT          NULL,
      employee_status       VARCHAR(200) NOT          NULL,
      supervisor               INT,

      PRIMARY KEY        (employee_id),
      CONSTRAINT         FK_SUPERVISOR FOREIGN KEY (supervisor) references
EMPLOYEE
);
```

<u>SERVICES Table</u>

```
create table SERVICES(
      service_id               INT                  NOT NULL,
      service_type            VARCHAR(200) NOT NULL,
      tenant_id                INT                  NOT NULL,
      start_time               DATETIME         NOT NULL,
      end_time                 DATETIME         NOT NULL,
      service_status          VARCHAR(200) NOT NULL,
      employee_signoff        BIT NOT NULL,
      employee_id             INT,

      PRIMARY KEY             (service_id),
      CONSTRAINT FK_SERVICES_EMPLOYEE FOREIGN KEY (employee_id)
      references EMPLOYEE(employee_id),
      CONSTRAINT FK_SERVICES_TENANT FOREIGN KEY(tenant_id) references
      TENANT(tenant_id)
);
```

<u>MAINTENANCE Table</u>

```
create table MAINTENANCE (
        request_id INT NOT NULL,
        apartment_id INT NOT NULL,
        request_status VARCHAR(200) NOT NULL,
        request_type VARCHAR(200) NOT NULL,
        request_raised_by INT NOT NULL,
        assigned_to INT,

        PRIMARY KEY (request_id)
        CONSTRAINT
        FK_MAINTENANCE_TENANT FOREIGN
        KEY(request_raised_by) references
        TENANT(tenant_id)
        );
```

<u>LEASE Table</u>

```
create table LEASE (
        lease_id        INT     NOT     NULL,
        apartment_id    INT     NOT     NULL,
        employee_id     INT     NOT     NULL,
        lease_status    VARCHAR(200)    NOT     NULL,
        lease_start_date DATE           NOT     NULL,
        lease_end_date  DATE,
        renewal_status  VARCHAR(200),
        renewed_from    INT,
        rent            INT NOT NULL,
        deposit         INT NOT NULL,
        processing_fees INT NOT NULL,

        PRIMARY KEY (lease_id),
        CONSTRAINT FK_LEASE_APARTMENT FOREIGN KEY (apartment_id)
        references APARTMENT(apartment_id),
        CONSTRAINT FK_LEASE_EMPLOYEE FOREIGN KEY (employee_id) references
        EMPLOYEE(employee_id),
        CONSTRAINT FK_RENEWAL FOREIGN KEY (renewed_from) references LEASE
);
```

PARKING Table

```
create table PARKING (
   parking_id              INT NOT NULL ,
   apartment_id            INT NOT NULL,
   parking_type            VARCHAR(200) NOT NULL,
   PRIMARY KEY             (parking_id),
   constraint              FK_PARKING FOREIGN KEY (apartment_id) REFERENCES
                   APARTMENT(apartment_id)
);
```

DATA INGESTION

Apartment Table

```
  insert into APARTMENT
        (apartment_id,building_no,lease_status,apartment_type,apartment_area,
         electricity_supply,mailbox,lease_id)
         values
              (3313,33,'ACTIVE','2B2B',1500,'TATA',33131,120305),
              (0817,08,'ACTIVE','1B1B',650,'TATA',08171,120301),
              (0117,01,'ACTIVE','1B2B',750,'ADANI',01171,120302),
              (0214,02,'ACTIVE','2B1B',850,'BSES',02141,120303),
              (1301,13,'INACTIVE','2B2B',1250,'RELIANCE',1301,120308),
              (0409,04,'ACTIVE','2B2B',1350,'TATA',0409,120304),
              (0511,05,'ACTIVE','1B1B',650,'GREEN',05111,120307),
              (0621,06,'EXPIRED','2B2B',1352,'ADANI',06211,120314),
              (0721,07,'ACTIVE','1B1B',700,'RELIANCE',07211,120315),
              (0918,09,'INACTIVE','2B1B',1132,'TATA',09181,120316),
              (0901,09,'ACTIVE','1B1B',650,'BSES',09011,120317),
              (1021,10,'ACTIVE','2B2B',1250,'RELIANCE',1021,120318),
              (1122,11,'INACTIVE','2B1B',1352,'GREEN',11221,120319);
```

Tenant Table

```
  insert into TENANT
        (tenant_id, tenant_name, tenant_status, permanent_address, contact_no,
  doc_id,nationality)
        values
        (1, 'Jeff Jackson','ACTIVE','Bellvue WA',9049919415,46,'USA'),
        (2, 'Bill Myers','ACTIVE','Seattle WA',9823481567,67,'USA'),
        (17, 'Melinda Piers','ACTIVE','Seattle WA',9323134117,13,'USA'),
        (9, 'Elon Anders','INACTIVE','Austin TX',9619046870,70,'USA'),
        (47, 'Mark Pickard','EXPIRED','Pao Alto CA',4694731826,26,'USA'),
        (48, 'Anil Sharma','ACTIVE','Bellvue WA',8176092856,56,'INDIA'),
        (19, 'Mukesh Mehra','ACTIVE','Mumbai IN',4699272833,33,'INDIA'),
        (15, 'Dahlia Hernandez','ACTIVE','Grand Prairie TX',4721547846,84,'USA'),
        (3, 'Bruno Mars', 'EXPIRED','Mumbai IN',4513684265,14,'USA'),
        (4, 'Edward Cooper', 'ACTIVE','London UK',5481621665,15,'GREAT BRITAIN'),
        (63, 'Marcus Brown', 'INACTIVE','Melbourne',61674236874,18,'AUSTRAILIA'),
        (42, 'Fernando Bielsa', 'ACTIVE','Buenos Aires AR',5933495321,34,'ARGENTINA'),
        (57, 'Joe Pratt', 'ACTIVE','Detroit MI',6413884265,82,'USA'),
        (37, 'Matt Harris', 'INACTIVE','Pittsburgh PA',5412235548,42,'USA');
```

Cohort Table

```
  insert into COHORT (tenant_id,apartment_id,isPrimary)
        values
        (15,3313,1),
        (1,0117,1),
        (2,0214,0),
        (17,0214,1),
        (47,0409,1),
        (48,0409,1),
        (19,0511,1),
        (3,0621,1),
        (4,0721,1),
        (63,0918,1),
        (42,0901,1),
        (57,1021,1),
        (37,1122,1);
```

Concierge Table

```
  insert into CONCIERGE (order_id,apartment_id,shipping_co,sender,box_no)
        values
                (86,3313,'USPS','Ravikiran',12),
                (17,3313,'UPS','Aaron',8),
                (03,0117,'DHL','Apple',12),
                (134,0409,'FedEx','David Finch',36),
                (91,1301,'Prime','Harry Styles',34),
                (07,0817,'SkyDel','Sangram',17),
                (11,0511,'USPS','Saurabh',21),
                (01,1122,'SkyDel','Ross',17),
                (42,1021,'USPS','Monica',9),
                (15,0918,'FedEx','Rachel',18);
```

Employee Table

```
insert into EMPLOYEE
        (employee_id,employee_name,contact_no,employee_address,designation,
joining_date, end_date,salary,employee_status,supervisor)
        values
                (1011,'Rohan Bacchhan',8462679089,'Frisco TX','General Manager','2017-11-
26',NULL,180230,'ACTIVE',NULL),
                (123,'Jackson Michael',1243486789,'Richardson TX','Growth Manager','2015-
09-26',NULL,120000,'ACTIVE',1011),
                (3711,'Phillip Morris',5135715673,'Allen TX','Leasing Manager','2020-09-
26',NULL,840210,'ACTIVE',1011),
                (671,'Sarah McCallan',4735125546,'Richardson TX','Community
Manager','2015-05-15',NULL,72300,'ACTIVE',123),
                (262,'Jeff Gomes',658184347,'Duncanville TX','Maintenance Manager','2010-
01-13','2020-04-09',62500,'INACTIVE',671),
                (261,'Keiran Jones',237568439,'Grapevine TX','Maintenance Manager','2020-
06-15',NULL,'67500','ACTIVE',671),
                (553,'Bruno Rodriguez',8795486789,'Plano TX','Leasing Officer','2018-03-
12',NULL,76500,'ACTIVE',3711),
                (420,'Brian Adams',8456881789,'Plano TX','Leasing Officer','2019-03-
12',NULL,76500,'ACTIVE',3711),
                (613,'Maria Fernandes',4716384923,'Plano TX','Leasing Officer','2014-06-
04','2021-07-27',71400,'INACTIVE',3711),
                (341,'Dave Henderson',6793657846,'Dallas TX','Community Officer','2013-07-
15','2019-02-05',66015,'INACTIVE',671),
                (141,'Mark Henry',263568546,'Greenville TX','Community Officer','2018-05-
15',NULL,63500,'ACTIVE',671),
                (841,'Johny Harris',753568347,'Dallas TX','Plumbing Officer','2020-06-
15',NULL,53500,'ACTIVE',261),
                (324,'David Powell',612568743,'Copell TX','Power Officer','2013-06-
15',NULL,57500,'ACTIVE',261),
                (314,'Josh Homes',6575184762,'Duncanville TX','Hygiene Officer','2004-06-
15',NULL,60500,'ACTIVE',261),
                (563,'Aaron Kane',5713267512,'Plano TX','Security Officer','2004-06-
15','2018-05-30',53500,'INACTIVE',262),
                (543,'Steve Mitchell',4537967471,'Frisco TX','Security Officer','2018-04-
15',NULL,56312,'ACTIVE',671),
                (473,'Connor Hicks',4863157956,'Argyle TX','Relationship Manager','2016-09-
26',NULL,88400,'ACTIVE',123),
                (469,'Collin Joshua',4367812354,'Hurst TX','Marketing Officer','2015-09-
26',NULL,80000,'ACTIVE',123);
```

<u>Lease Table</u>

```
  insert into LEASE
         (lease_id,apartment_id,employee_id,lease_status,lease_start_date,
                 lease_end_date,renewal_status,renewed_from,rent,deposit,processing_fees)
  values
         (120300,409,473,'EXPIRED','2016-06-20','2017-07-
  23','RENEWED',NULL,1200,200,50),
         (120301,409,473,'EXPIRED','2017-07-23','2018-07-23',NULL,120300,1320,200,50),
         (120302,117,420,'ACTIVE','2018-06-20','2021-12-31',NULL,NULL,1500,250,75),
         (120310,214,123,'EXPIRED','2018-05-13','2020-09-
  14','RENEWED',NULL,1650,250,85),
         (120303,214,3711,'ACTIVE','2020-09-30','2022-07-14',NULL,120310,1850,250,85),
         (120308,1301,473,'INACTIVE','2021-12-31','2022-01-
  01',NULL,NULL,1550,200,50),
         (120305,3313,671,'ACTIVE','2021-05-14','2022-10-07',NULL,NULL,1400,150,50),
         (119900,511,553,'EXPIRED','2020-01-13','2021-04-
  29','RENEWED',NULL,1320,200,50),
         (120307,511,553,'ACTIVE','2021-05-30','2023-05-30',NULL,119900,1260,200,50),
         (120304,409,1011,'ACTIVE','2020-10-20','2023-12-23',NULL,NULL,1900,300,80),

         (120314,621,3711,'EXPIRED','2020-09-30','2021-08-13',NULL,NULL,1850,250,80),
         (120320,721,473,'EXPIRED','2020-09-30','2021-08-
  13','RENEWED',NULL,900,200,50),
         (120315,721,473,'ACTIVE','2020-10-20','2022-01-01',NULL,120320,900,200,50),
         (120316,918,420,'INACTIVE','2022-02-12','2023-12-23',NULL,NULL,1900,300,80),
         (120317,901,1011,'ACTIVE','2020-10-20','2021-12-23',NULL,NULL,900,200,50),
         (120318,1021,1011,'ACTIVE','2019-04-29','2023-12-23',NULL,NULL,1900,300,80),
         (120319,1122,553,'INACTIVE','2020-02-29','2022-03-
01',NULL,NULL,1900,300,80);
```

<u>Maintenance Table</u>

```
  insert into MAINTENANCE
         (request_id,apartment_id,request_status,request_type,request_raised_by,assigned_to)
  values
         (1533,117,'PENDING','PLUMBING',1,841),
         (1501,117,'COMPLETED','POWER WORKS',1,324),
         (1543,1301,'PENDING','POWER WORKS',9,324),
         (1512,409,'PENDING','PLUMBING',47,841),
         (1515,511,'COMPLETED','PLUMBING',19,841);
```

Parking Table

```
  insert into PARKING
          (parking_id,apartment_id,parking_type)
  values
          (1171,117,'GARAGE'),
          (1172,117,'SHED'),
          (2141,214,'GARAGE'),
          (2142,214,'GARAGE'),
          (13011,1301,'GARAGE'),
          (13012,1301,'SHED'),
          (5111,511,'SHED'),
          (9011,901,'SHED'),
          (10211,1021,'GARAGE'),
          (10212,1021,'SHED');
```

Documents Table

```
  insert into DOCUMENTS
          (tenant_id,doc_id,documents)
          values
                  (1,46,'Income Proof,Driving License'),
                  (2,67,'Income Proof,Passport'),
                  (9,70,'Income Proof,SSN'),
                  (15,84,'Income Proof,Driving License'),
                  (17,13,'Income Proof,Passport'),
                  (19,33,'VISA,Passport'),
                  (47,26,'Income Proof,SSN'),
                  (48,56,'VISA,Passport'),
                  (3,14,'Income Proof,SSN'),
                  (4,15,'VISA,Passport'),
                  (63,18,'VISA,Passport'),
                  (42,34,'VISA,Passport'),
                  (57,82,'Income Proof,SSN'),
                  (37,42,'Income Proof,SSN');
```

Services Table

```
insert into SERVICES
        (service_id,service_type,tenant_id,start_time,end_time,service_status,employee_signo
ff,employee_id)
        values
                (89,'Pool',15,'2021-10-10 09:00:00','2021-10-10 09:30:00','Done',1,141),
                (95,'Game Room',19,'2021-10-24 18:00:00','2021-10-24
21:00:00','Approved',1,671),
                (99,'Conference Room',48,'2021-10-28 15:00:00','2021-10-28
16:00:00','Received',0,NULL),
                (97,'Study Area',2,'2021-10-22 19:30:00','2021-10-22 20:30:00','In
Progress',1,473),
                (98,'Game Room',17,'2021-10-12 13:00:00','2021-10-12
14:30:00','Done',1,473);
```

SQL Queries and Use-Case Reports

Scenario 1
Find building number, apartment type, tenant status and primary status of tenants present in all apartments

Approach
Joining the following 3 tables will result in expected data output.
1. Tenant
2. Apartment
3. Cohort

SQL query :
```
select tenant.tenant_id, tenant_name,  tenant_status, apartment.apartment_id,
       apartment.building_no,
       apartment.apartment_type,
       (case when COHORT.isPrimary=1 then 'Primary' else 'Other' END)
Primary_tenant_status
       from tenant, apartment,cohort
       where tenant.tenant_id = cohort.tenant_id
       and cohort.apartment_id = apartment.apartment_id;
```

Data Output:

tenant_id	tenant_name	tenant_status	apartment_id	building_no	apartment_type	Primary_tenant_status
1	Jeff Jackson	ACTIVE	117	1	1B2B	Primary
2	Bill Myers	ACTIVE	214	2	2B1B	Other
3	Bruno Mars	EXPIRED	621	6	2B2B	Primary
4	Edward Cooper	ACTIVE	721	7	1B1B	Primary
15	Dahlia Hernandez	ACTIVE	3313	33	2B2B	Primary
17	Melinda Piers	ACTIVE	214	2	2B1B	Primary
19	Mukesh Mehra	ACTIVE	511	5	1B1B	Primary
37	Matt Harris	INACTIVE	1122	11	2B1B	Primary
42	Fernando Bielsa	ACTIVE	901	9	1B1B	Primary
47	Mark Pickard	EXPIRED	409	4	2B2B	Primary
48	Anil Sharma	ACTIVE	409	4	2B2B	Primary
57	Joe Pratt	ACTIVE	1021	10	2B2B	Primary
63	Marcus Brown	INACTIVE	918	9	2B1B	Primary

Scenario 2:
Display the count of total apartments and group them by electricity suppliers. List the suppliers having apartment count more than 2.

SQL query:
```
select electricity_supply,count(apartment_id) No_of_Apartments
from APARTMENT
group by electricity_supply
having count(apartment_id)>2
```

Data Output:

electricity_supply	No_of_Apartments
RELIANCE	3
TATA	4

Scenario 3:
Group the Number of active employees and their salary by designation level.

SQL query:
```
select designation, COUNT(EMPLOYEE_ID) Employee_count,sum(salary) Total_Salary
from EMPLOYEE
where employee_status='ACTIVE'
GROUP BY designation
```

Data Output:

designation	Employee_count	Total_Salary
Community Manager	1	72300
Community Officer	1	63500
General Manager	1	180230
Growth Manager	1	120000
Hygiene Officer	1	60500
Leasing Manager	1	840210
Leasing Officer	2	153000
Maintenance Manager	1	67500
Marketing Officer	1	80000
Plumbing Officer	1	53500
Power Officer	1	57500
Relationship Manager	1	88400
Security Officer	1	56312

Scenario 4:
Find the tenants who have not used a single service yet

Approach:
These 4 tables have been considered and Left Outer Join has been applied to find out which tenants haven't used a single service yet.
1. Tenant
2. Apartment
3. Cohort
4. Services

SQL query:

```
        select A.* from (select
        tenant.tenant_id,tenant_name,apartment.apartment_id,apartment.building_no
        from tenant, apartment,cohort
        where tenant.tenant_id = cohort.tenant_id
        and cohort.apartment_id = apartment.apartment_id ) a LEFT OUTER join services b
        on a.tenant_id=b.tenant_id
        WHERE b.tenant_id IS NULL;
```

Data Output:

tenant_id	tenant_name	apartment_id	building_no
1	Jeff Jackson	117	1
3	Bruno Mars	621	6
4	Edward Cooper	721	7
37	Matt Harris	1122	11
42	Fernando Bielsa	901	9
47	Mark Pickard	409	4
57	Joe Pratt	1021	10
63	Marcus Brown	918	9

Scenario 5:
Tenants who have raised maintenance requests and are still in pending condition.

Approach:
These 3 tables have been considered and joined to find information about pending requests.
1.Tenant
2. Cohort
3. Maintenance

SQL query:

```
        select
        tenant.tenant_id,tenant_name,COHORT.apartment_id,MAINTENANCE.request_id,REQUEST
        _STATUS,request_type
              from tenant,cohort,MAINTENANCE
              where tenant.tenant_id = cohort.tenant_id
              and MAINTENANCE.apartment_id=COHORT.apartment_id
              and request_status='PENDING';
```

Data Output:

tenant_id	tenant_name	apartment_id	request_id	REQUEST_STATUS	request_type
1	Jeff Jackson	117	1533	PENDING	PLUMBING
47	Mark Pickard	409	1512	PENDING	PLUMBING
48	Anil Sharma	409	1512	PENDING	PLUMBING

Scenario 6
Find the active employees and their respective supervisors' details who have joined after 2014

Approach:

Employee table has been used to find Employees and their supervisors' details. Employee details have been self-joined again with the employee table to find respective supervisor details.

SQL query:

```
select a.employee_id,a.employee_name,a.contact_no,a.employee_address,b.employee_name
supervisor_name, b.contact_no supervisor_contact_no,b.employee_address
supervisor_address from
(select employee_id,employee_name,contact_no,employee_address,supervisor from EMPLOYEE
where employee_status='ACTIVE'
and joining_date>'2014-12-31') a, employee b
where a.supervisor=b.employee_id;
```

Data Output:

employee_id	employee_name	contact_no	employee_address	supervisor_name	supervisor_contact_no	supervisor_address
123	Jackson Michael	1243486789	Richardson TX	Rohan Bacchhan	8462679089	Frisco TX
141	Mark Henry	263568546	Greenville TX	Sarah McCallan	4735125546	Richardson TX
261	Keiran Jones	237568439	Grapevine TX	Sarah McCallan	4735125546	Richardson TX
420	Brian Adams	8456881789	Plano TX	Phillip Morris	5135715673	Allen TX
469	Collin Joshua	4367812354	Hurst TX	Jackson Michael	1243486789	Richardson TX
473	Connor Hicks	4863157956	Argyle TX	Jackson Michael	1243486789	Richardson TX
543	Steve Mitchell	4537967471	Frisco TX	Sarah McCallan	4735125546	Richardson TX
553	Bruno Rodriguez	8795486789	Plano TX	Phillip Morris	5135715673	Allen TX
671	Sarah McCallan	4735125546	Richardson TX	Jackson Michael	1243486789	Richardson TX
841	Johny Harris	753568347	Dallas TX	Keiran Jones	237568439	Grapevine TX
3711	Phillip Morris	5135715673	Allen TX	Rohan Bacchhan	8462679089	Frisco TX

Scenario 7:
Display the tenants and their apartment details who have submitted 'SSN' as one of their document proofs. Tenants who are not currently associated with apartment system should be excluded from result set.

Approach:
These 4 tables have been considered and joined to find information about Tenant's documents.
1. Tenant
2. Cohort
3. Apartment
4. Documents

SQL query:

```
Select
a.tenant_id,a.tenant_name,a.tenant_status,b.apartment_id,b.building_no,d.documents
from TENANT a,APARTMENT b,COHORT c,DOCUMENTS d
where a.tenant_id=c.tenant_id
and b.apartment_id=c.apartment_id
and a.tenant_id=d.tenant_id
and tenant_status <> 'EXPIRED'
and DOCUMENTS like '%SSN%'
```

Data Output:

tenant_id	tenant_name	tenant_status	apartment_id	building_no	documents
37	Matt Harris	INACTIVE	1122	11	Income Proof,SSN
57	Joe Pratt	ACTIVE	1021	10	Income Proof,SSN

Contributions

Name	Contribution
Ravikiran Ramchandra Pise	• Data Architecture • Data Modelling • ER Diagram
Sangram Tushar Mohite Patil	• Implementation of SQL queries and verification • ER Diagram
Chinmay Mukesh Sinari	• Data Ingestion • Report Creation
Tejas Kadu	• Data Validation • Report Creation