

Conceptual Database Design

1. Historical Membership Tier

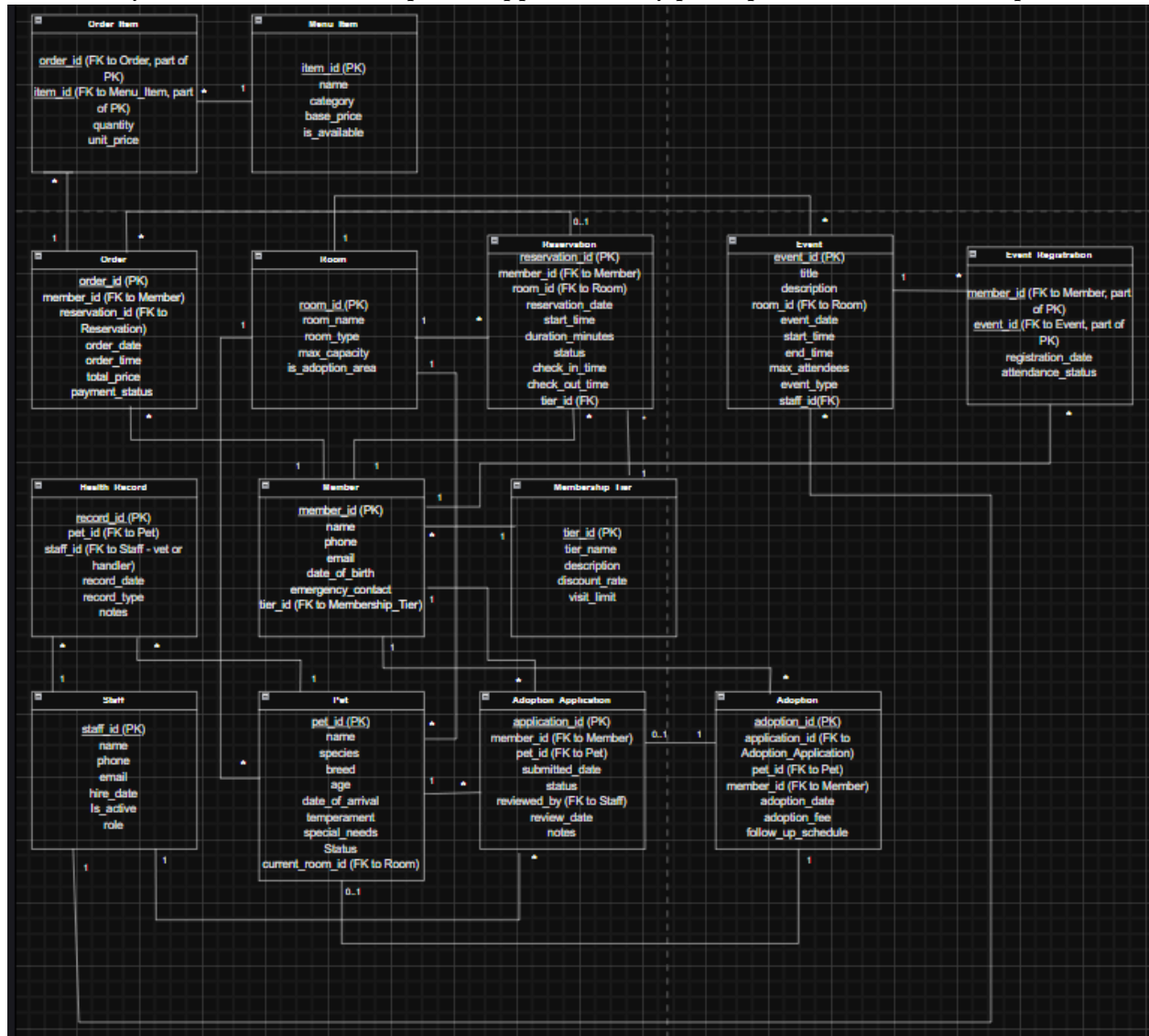
Although each Member has a current membership tier, the Reservation entity stores the tier at the time of the visit. This ensures accurate reporting when membership changes over time (Query 2).

2. Adoption Rules

Every Adoption must reference exactly one Adoption_Application and exactly one Pet.

A Pet or Application may exist without an adoption, but an Adoption cannot exist without both.

Additionally, each Pet and each Adoption_Application may participate in at most one Adoption.



Logical Database Design

After we completed our ER Diagram, we had to convert it into actual tables in Oracle's SQL. The schema of our tables can be seen in the tables below:

Membership_Tier

<i>tier_id</i>	INTEGER
tier_name	VARCHAR(50) NOT NULL
description	VARCHAR(200)
discount_rate	NUMBER(3) (BETWEEN 0 AND 100)
visit_limit	INTEGER

Room

<i>room_id</i>	INTEGER
room_name	VARCHAR(100) NOT NULL
room_type	VARCHAR(50)
max_capacity	INTEGER NOT NULL
is_adoption_area	NUMBER(1) (Must be 0 or 1)

Member

<i>member_id</i>	INTEGER
name	VARCHAR2(100) NOT NULL
phone	VARCHAR(20)
email	VARCHAR2(100)
date_of_birth	DATE
emergency_contact	VARCHAR2(200)
tier_id	INTEGER (FK to Membership_Tier)

Reservation

<i>reservation_id</i>	INTEGER
member_id	INTEGER NOT NULL (FK to Member)
room_id	INTEGER NOT NULL (FK to Room)
start_time	DATE NOT NULL
duration_minutes	INTEGER NOT NULL
status	VARCHAR(20)
check_in_time	TIMESTAMP NOT NULL
check_out_time	TIMESTAMP NOT NULL
tier_id	INTEGER (FK to Membership_Tier)

Staff

<i>staff_id</i>	INTEGER
name	VARCHAR2(100) NOT NULL
phone	VARCHAR(20)
email	VARCHAR2(100)
hire_date	DATE
is_active	NUMBER(1) (Must be 0 or 1)
role	VARCHAR2(50)

Health_Record

<i>record_id</i>	INTEGER
pet_id	(FK to Pet)
staff_id	INTEGER NOT NULL (FK to Staff)
record_date	DATE NOT NULL
record_type	VARCHAR2(50)
notes	VARCHAR2(500)

Pet

<i>pet_id</i>	INTEGER
name	VARCHAR2(100) NOT NULL
species	VARCHAR2(100)
breed	VARCHAR2(100)
age	INTEGER
date_of_arrival	DATE
temperament	VARCHAR2(200)
special_needs	VARCHAR2(200)
status	VARCHAR2(30)
current_room_id	INTEGER (FK to Room)

Adoption_Application

<i>application_id</i>	INTEGER
member_id	INTEGER NOT NULL (FK to Member)
pet_id	INTEGER NOT NULL (FK to Pet)
submitted_date	DATE NOT NULL
status	VARCHAR2(20)
reviewed_by	INTEGER (FK to Staff)
review_date	DATE
notes	VARCHAR(500)

Adoption

<i>adoption_id</i>	INTEGER
application_id	INTEGER NOT NULL (FK to Adoption_Application)
pet_id	INTEGER NOT NULL (FK to Pet)
member_id	INTEGER NOT NULL (FK to Member)
adoption_date	DATE
adoption_fee	NUMBER(7, 2)
follow_up_schedule	DATE

Menu_Item

<i>item_id</i>	INTEGER
name	VARCHAR(100) NOT NULL
category	VARCHAR(50)
base_price	NUMBER(6, 2) NOT NULL
is_available	NUMBER(1) (Must be 0 or 1)

Customer_Order

<i>order_id</i>	INTEGER
member_id	INTEGER (FK to Member)
reservation_id	INTEGER (FK to Reservation)
order_date	DATE
order_time	TIMESTAMP
total_price	NUMBER(8, 2)
payment_status	VARCHAR2(20)

Order_Item

<i>order_id</i>	INTEGER (FK to Customer_Order)
<i>item_id</i>	INTEGER (FK to Menu_Item)
quantity	INTEGER
unit_price	NUMBER(7,2) NOT NULL

Event

<i>event_id</i>	INTEGER
title	VARCHAR(100)
description	VARCHAR(400)
room_id	INTEGER NOT NULL (FK to Room)
event_date	DATE NOT NULL
start_time	TIMESTAMP
end_time	TIMESTAMP
max_attendees	INTEGER NOT NULL
event_type	VARCHAR2(50)
staff_id	INTEGER NOT NULL (FK to Staff)

Event_Registration

<i>member_id</i>	INTEGER (FK to Member)
<i>event_id</i>	INTEGER (FK to Event)
registration_date	DATE
attendance_status	VARCHAR(30)

Normalization Analysis

All of the FDs are as follows:

- Order_Item(order_id, item_id) → quantity, unit_price
- Menu_Item(item_id) → name, category, base_price, is_available
- Customer_Order(order_id) → member_id, reservation_id, order_date, order_time, total_price, payment_status
- Room(room_id) → room_name, room_type, max_capacity, is_adoption_area
- Reservation(reservation_id) → member_id, room_id, reservation_date, start_time, duration_minutes, status, check_in_time, check_out_time, tier_id
- Event(event_id) → title, description, room_id, event_date, start_time, end_time, max_attendees
- Event_Registration(member_id, event_id) → registration_date, attendance_status
- Health_Record(record_id) → pet_id, staff_id, record_date, record_type, notes
- Membership_Tier(tier_id) → tier_name, description, discount_rate, visit_limit
- Staff(staff_id) → name, phone, email, hire_date, is_active, role
- Member(member_id) → name, phone, email, date_of_birth, emergency_contact, tier_id
- Pet(pet_id) → name, species, breed, age, date_of_arrival, temperament, special_needs, status, current_room_id
- Adoption_Application(application_id) → member_id, pet_id, submitted_date, status, reviewed_by, review_date, notes
- Adoption(adoption_id) → application_id, pet_id, member_id, adoption_date, adoption_fee, follow_up_schedule

In order to know if our tables are in 3NF / BCNF, we must first look at the definitions of 1NF and 2NF. First of all, the tables are clearly in 1NF form because none of our data is in set form, meaning each entry under a column only contains the one value. We also know that the tables are in 2NF because of our two tables that have multiple candidate keys Order_Item and Event_Registration.

We know that Order_Item relies on both the order_id and item_id because you need both items to be able to uniquely identify an order since an item can show up in multiple orders and an order can have multiple items. We also need both the member_id and event_id to uniquely identify an event registration since a member can go to multiple events and an event has multiple members that can attend.

Lastly, to ensure we are in 3NF / BCNF we know that none of our tables have transitive dependencies. All of our non-primary key attributes only depend on primary keys.

Query Description

Query 1 - Adoption Applications for a Pet

This query lists all adoption applications submitted for a specific pet.

It joins Adoption_Application, Member, and Staff to show:

- Applicant name
- Application date
- Current status
- Assigned coordinator

This helps staff track interest and monitor adoption progress.

Query 2 - Visit History for a Member

This query provides a complete history of a member's visits and spending.

It joins Reservation, Room, Membership_Tier, and Customer_Order to display:

- Reservation date and time
- Room visited
- Membership tier at the time
- Total spending for each visit

This supports customer analysis and billing review.

Query 3 - Upcoming Events with Available Capacity

This query identifies upcoming events that are not yet full.

It combines Event, Event_Registration, Room, and Staff to report:

- Event name, date, and time
- Room location
- Current number of attendees vs. capacity
- Event coordinator

Useful for planning and promoting events.

Query 4 - Custom Query: Top Members by Spending

This query ranks high-value members based on total spending.

It joins Member, Membership_Tier, Reservation, and Customer_Order, and filters by a user-entered minimum spending amount.

The output includes:

- Member ID and name
- Tier
- Number of visits
- Total spending

This helps management understand customer value and behavior.