Ski Resort Database Design

GROUP 14: CONNOR, LUIS, MOHAMMAD, NATHAN

Spring 2025

Contents

§1 Conceptual database design

Figure 1 shows the E–R diagram of the ski resort database. To design the conceptual schema of the database we first did a detailed walk through fot the program specifications to record what tables are going to be needed to store data. The ER diagram we made captures the ski resorts day to day operations tracking members and their activites as well as the resorts staff and income via its services. Attributes are constrained via sql datatypes.

§2 Logical database design

Below is the final relational schema derived from the E–R diagram. Each table lists its attributes, primary keys (**PK**), and foreign keys (**FK**) explicitly. This design reflects a direct and normalized translation of the conceptual model into the relational model used in Oracle SQL.

- **Property**(<u>propertyID</u>, name, address, propertyType)
- **Shop**(<u>shopID</u>, name, shopType, buildingID^{FK}, income)
- **Equipment**(<u>equipmentID</u>, equipmentType, equipmentSize, archived)
- **Rental**(<u>rentalID</u>, equipmentID^{FK}, passID^{FK}, rentalTime, returnStatus)
- **Pass**(<u>passID</u>, memberID^{FK}, numUses, passType, price, exprDATE)
- **LiftLog**(passID^{FK}, liftID^{FK}, dateTime)
- **Lift**(<u>liftID</u>, liftName, openTime, closeTime, abilityLevel, status)

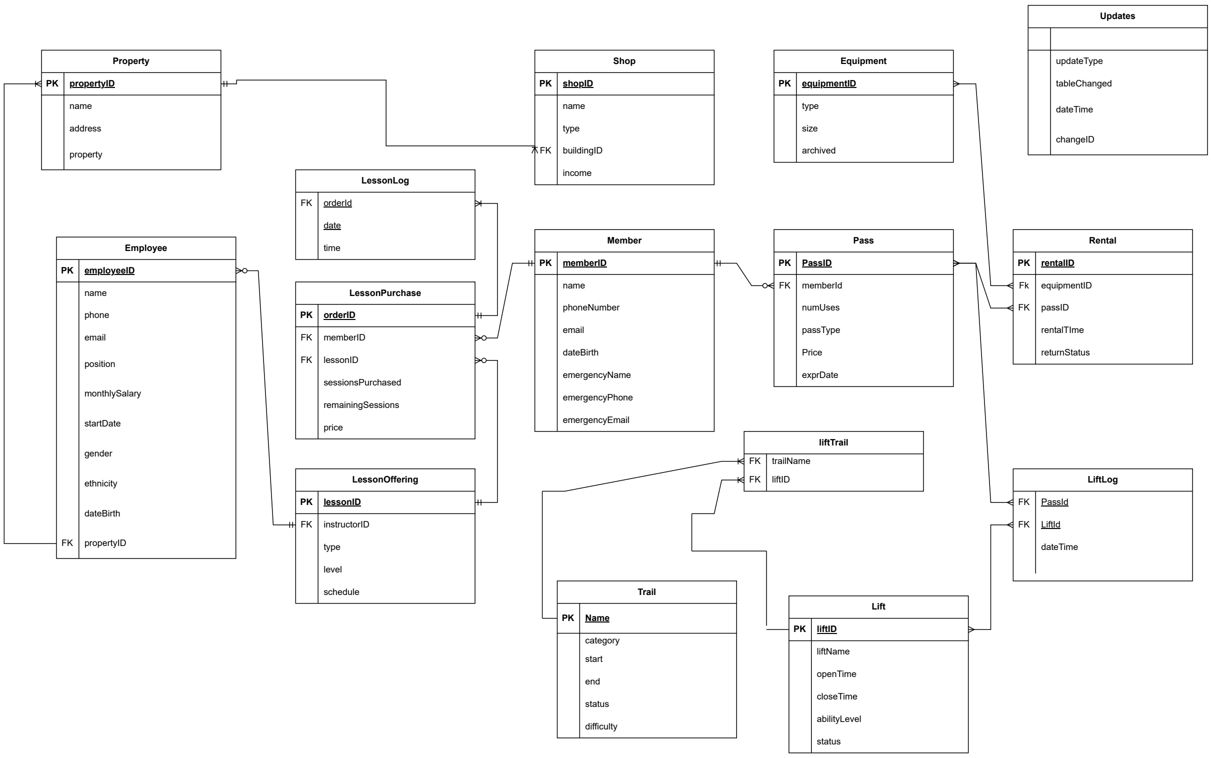

Figure 1: E–R Diagram of the Ski Resort Database

- **TrailLift**(liftID$^{\text{FK}}$, trailName$^{\text{FK}}$)
- **Trail**(<u>name</u>, category, startPos, endPos, status, difficulty)
- **Member**(<u>memberID</u>, name, phoneNumber, email, dateBirth, emergencyName, emergencyPhone, emergencyEmail)
- **LessonLog**(orderID$^{\text{FK}}$, dateTime)
- **LessonPurchase**(<u>orderID</u>, memberID$^{\text{FK}}$, lessonID$^{\text{FK}}$, sessionsPurchased, remainingSessions, price)
- **LessonOffering**(<u>lessonID</u>, instructorID$^{\text{FK}}$, lessonType, skillLevel, schedule)
- **Employee**(<u>employeeID</u>, name, phone, email, position, monthlySalary, startDate, gender, ethnicity, dateBirth, propertyID$^{\text{FK}}$)
- **Updates**(updateType, tableChanged, changeID, dateTime)

§3 Normalization analysis

These all adhere to 1NF as all set attributes are non set values. Also, they all adhere to 2NF because all non-prime attributes are fully functionally dependant on the candidate keys.

Property(propertyID, name, address, propertyType) FDs: propertyID -¿ name, propertyID -¿ adress, propertyID -¿ propertyType This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Shop(shopID, name, type, buildingID, income) FDs: shopID -¿ name, shopID -¿ type, shopID -¿ buildingID, shopID -¿ income This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Equipment(equipmentID, type, size, archived) FDs: equipmentID -¿ type, equipmentID -¿ size, equipmentID -¿ archived This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Rental(rentalID, equipmentID, passID, rentalTime, returnStatus) FDs: rentalID -¿ equipmentID, rentalID -¿ passID, rentalID -¿ rentalTime, rentalID -¿ rentalStatus This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Pass(passID, memberID, numUses, passType, price, exprDate) FDs: passID -¿ memberID, passID -¿ numUses, passID -¿ passType, passID -¿ price, passID -¿ exprDate This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

LiftLog(passIDFK, liftID, dateTime) FDs: passID -¿ liftID, passID -¿ dateTime This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Lift(liftID, liftName, openTime, closeTime, abilityLevel, status) FDs: liftID -¿ liftName, liftID -¿ openTime, liftID -¿ closeTime, liftID -¿ abilityLevel, liftID -¿ status This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

TrailLift(liftID, trailName) FDs: liftID -¿ trailName This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Trail(name, category, start, end, status, difficulty) FDs: name -¿ category, name -¿ start, name -¿ end, name -¿ status, name -¿ difficulty This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Member(memberID, name, phoneNumber, email, dateBirth, emergencyName, emergencyPhone, emergencyEmail) FDs: memberID -¿ name, memberID -¿ phoneNumber, memberID -¿ email, memberID -¿ dataBirth, memberID -¿ emergencyName, memberID -¿ emergencyPhone, memberID -¿ emergencyEmail This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

LessonLog(orderID, dateTime) FDs: orderID -¿ dateTime This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

LessonPurchase(orderID, memberID, lessonID, sessionsPurchased, remainingSessions, price) FDs: orderID -¿ memberID, orderID -¿ lessonID, orderID -¿ sessionsPurchased, orderID -¿ remainingSessions, orderID -¿ price This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

LessonOffering(lessonID, instructorID, type, level, schedule) FDs: lessonID -¿ instructorID, lessonID -¿ type, lessonID -¿ level, lessonID -¿ schedule This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Employee(employeeID, name, phone, email, position, monthlySalary, startDate, gender, ethnicity, dateBirth, propertyIDFK) FDs: employeeID -¿ name, employeeID -¿ phone, employeeID -¿ email, employeeID -¿ position, employeeID -¿ monthlySalary, employeeID -¿ startDate This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

Updates(updateType, tableChanged, changeID, dateTime) FDs: dateTime -¿ updateType, dateTime -¿ tableChanged, dateTime -¿ changeID This is in BCNF as for all of the FDs X -¿ Y, X is a superkey (it is the PK)

§4 Query description

§4.1 Custom Query: Monthly Income Summary

Query Goal: Calculate the gross monthly income of the resort by subtracting total employee salaries from the sum of all incomes recorded across the properties.

Motivation: This query helps stakeholders monitor the profitability of the resort's operations, combining staff payroll and property performance in a single monthly snapshot.

Relations Involved:

- `Property`
- `Shop`
- `Employee`

Query Details: For each month, aggregate total income from properties (e.g., gift shops, rental centers), subtract the sum of salaries of all employees, and report the net income. This could be extended to include breakdowns by property type or department.