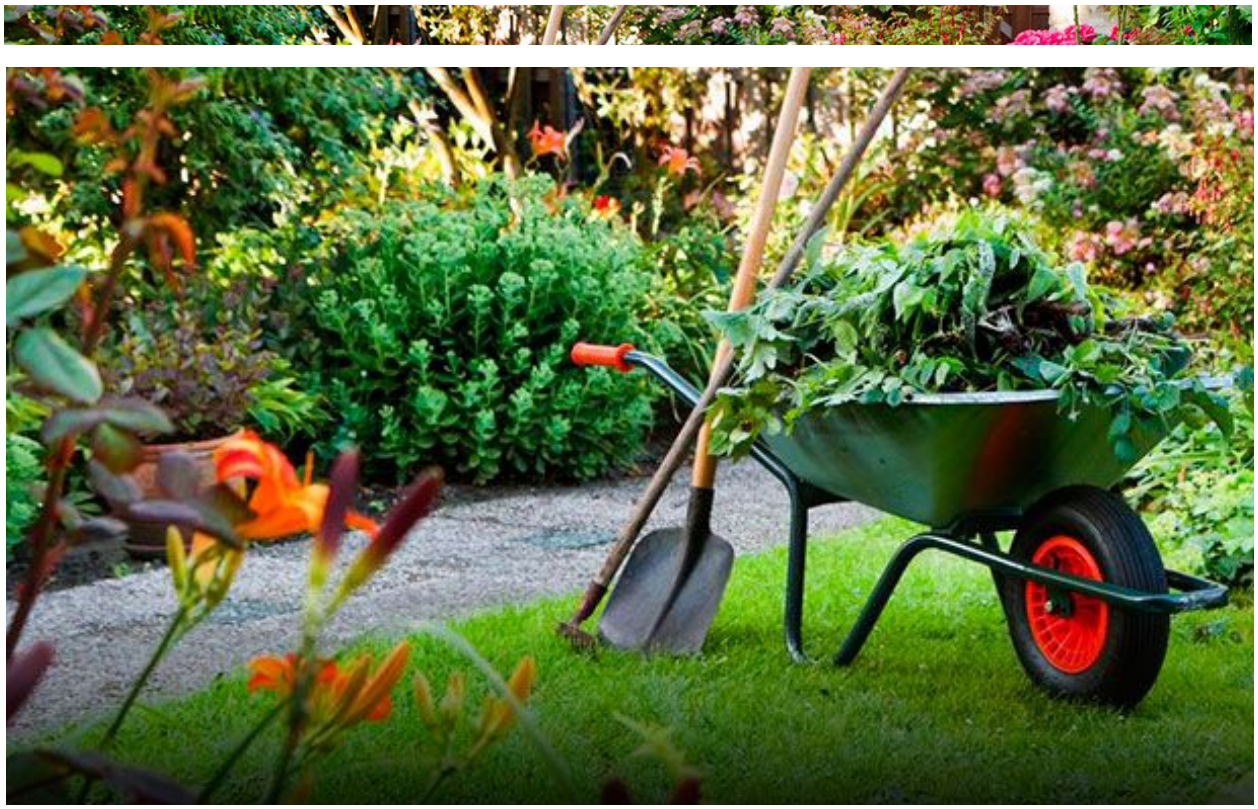


Landscape Service Database

Erick Toscano

Eddie Velasco

CMPS 3420 - Fall 2018

California State University, Bakersfield

Table of Contents

Phase 1: Fact-Finding, Information Gathering, and Conceptual Database Design

1.1 Fact-Finding Techniques and Information Gathering

In order to build the database for an organization, a database designer must have a thorough comprehension of the organization they are hired to design. The following section will give the introduction to the organization, an explanation of the research process, the scope of the database design, and itemized descriptions of the entity and relationship sets.

1.1.1 Introduction to Enterprise/Organization

Green Landscaping is a fictitious landscaping service designed to represent aspects of how Toscano Landscaping, a local landscaping business, handles their work flow. Green Landscaping offers landscaping services to residential homes on a weekly basis. Services are offered in the form of contracts with defined terms or one-time service work. The owner or manager of the business is in charge of keeping track of all contracts, expenses, income, and employee payroll.

1.1.2 Description of Fact-Finding Techniques

Fact-finding for the Green Landscaping database relied much on the general understanding of landscaping businesses as well as information provided by Toscano Landscaping which is owned and operated by an immediate family member. We verified facts and gathered information through general talks and inquiries of how Toscano Landscaping ran their business. This allowed us to design a database that fully captured the intended business model. By interviewing the appropriate and potential users of the database we formed a well-thought out structure for our database. From contracts to tools and so on, we could observe and conceptualize the regular logistics of how the business was managed.

1.1.3 Database Design Scope

An important aspect when developing the concept of a database is to know what segments of the organization will be represented by the database. We call the

real world aspects the miniworld or universe of discourse. This is important so we can keep in mind the potential users and applications our database will supply. Landscaping businesses keep written documents to track all the contracts, services, client information, and employee information. They also require employees to have access to routes, addresses, work vehicles, and tools. The goal for the database we will design for our business, Green Landscaping, will be to create an efficient and easy to use system for both owner and employee interactions. An effective model will enable us to implement features and applications that will help us in our goal.

1.1.4 Entity and Relationship Set Description

After all our information and facts are gathered, the data can be represented as entity sets and relationship sets as is described in a typical E-R model. In the remainder of this section, these entities and relationships will be described. A further detailed explanation will be provided in section 1.2.

Entity Type Definitions

Income: Income_ID, Date, Type, Amount, Description
An **income** is a record of a payment received to the business.

Contract: Contract_ID, Price, Description
A **contract** is a document written between the business and a client for services.

House: House_ID, Address, Start_Date, End_Date
A **house** is the centralized location for services that will be rendered to client.

Additional Service: Service_ID, Price, Description
An **additional service** is a service not specified under contract but performed for a house.

Client: Client_ID, Name, Address, Phone Number
A **client** is an individual who owns the house where services will be rendered.

Route: Route_ID, Start_Date, End_Date
A **route** is a designated list of houses in which employees will follow to perform services.

Car: Car_ID, Start_Date, End_Date, Car_Desc
A **car** is a vehicle used by an employee or employees to transport tools and perform services.

Tool: Tool_ID, Brand, Name, Description
A **tool** is a piece of equipment used by an employee kept in a car that will be used to landscape.

Employee: Employee_ID, Name, Address, Phone, Salary
An **employee** is an individual who works for the business, performs work, and is paid.

Payment: Payment_ID, Amount, Date, Type, Hrs_Worked, Description
A **payment** is a record of a payment given to an employee.

Relationship Type Definitions

Employee **paid** by Payment; Cardinality: 1..N; Participation: Total, Total

Employee **assigned to** Car; Cardinality: N..1; Participation: Total, Total

Car **contains** Tools; Cardinality: 1..N; Participation: Total, Total

Route **worked by** Car; Cardinality: 1..1; Participation: Total, Total

House **assigned** to Route; Cardinality: 1..1; Participation: Total, Total

House **owned by** Client; Cardinality: N..1; Participation: Partial, Total

Additional Service **provided to** House; Cardinality: N..1; Participation: Partial, Total

Income **produced by** Contract; Cardinality: N..1; Participation: Total, Total

1.2 Conceptual Database Design

In this section we will discuss the conceptual design to Green landscaping database. We will describe its entities and corresponding attributes and some operations that will be supported by this model.

Before we design a fully operational database we must first implement a conceptual design so we can understand how the data will be stored. In this example we will be using the Entity Relationship (ER) model to understand the behavior of our database.

In this model we will describe an entity as an object, such as employee or client; with these entities will be attributes which will describe qualities of the entity. We also create relationships between entities which will describe how these objects are associated in relation to the database.

We must understand this blueprint of the database, or schema, which should not change too often since it is a description of the database occurrences before they have been initiated, so we can firmly present a diagram that visually represents the conceptual design.

1.2.1 Entity Type Description

Entities describe real world-objects and are defined by their name and attributes they contain. For example in this in this model the most important entities are Employee, Client, Route. Now we will describe each entity name, attributes, domain, keys, candidate key.

Entity: Employee

Description: An Employee get paid by the landscaping company, also an Employee will be assign to a Car with two descriptive attributes start_date and end_date. Employee will work either as a driver-employee or just an employee, the distinction between them is that the one that drive earns more income.

Candidate key: employee_ID
Primary key: employee_ID
Strong/Weak Entity: Strong
Fields to be indexed: employee_Id, salary

Attribute name:	Employee_ID	Name	Address	Phone	Salary
Description	Unique id assigned to employees to track they everyday start - end work hours and their payment.	Name of the employee (First, Last)	Address of employee (123 Street St.)	Phone number to reach employee	The amount paid to each employee
Type	Integer	String	String	int	float
Value / Range	0-MaxID	any	any	000-000-0000 to 999-999-9999	0-9999
Null allowed	no	no	no	yes	yes
Unique	yes	no	no	yes	no
Simple / Composite	simple	composite	composite	simple	simple

Entity: Client

Description: A Client is most of the time the owner of the house on which the Landscaping company works on. Client may own many houses, Client may be a tenant. Client is also with whom the company signs the contract to start working on, and from whom the company gets its income.

Candidate key: client_ID
Primary key: client_ID
Strong/Weak Entity: Strong
Fields to be indexed: client_ID

Attribute name:	Client_ID	Name	Address	Phone
Description	Unique id assigned to clients to keep track of their houses and payments	Name of the client (First, Last)	Address of client (123 Street St.)	Phone number to reach client
Type	Integer	String	String	int
Value / Range	0-MaxID	any	any	000-000-0000 to 999-999-9999
Null allowed	no	no	no	yes
Unique	yes	no	no	yes
Simple / Composite Attributes	simple	composite	composite	simple
Single / Multi value	single	single	single	single

Entity: Income

Description: Income is a payment given to the company by the client in exchange for landscape services which might include extra services. Payment sometimes may be lower or higher than usual, for that we use description. Payments can be made via bank deposit, check or cash.

Candidate key: income_ID
Primary key: income_ID
Strong/Weak Entity: Strong
Fields to be indexed: income_ID, date, type, amount

Attribute name:	income_ID	date	type	amount	description
Description	Unique id assigned to each payment to keep track of them	- If cash, date the payment was received. - If check or money order, date on check. - If bank, date on payment received	Cash, check, money order, bank transaction	Amount each client pays for Landscape service	To record if there is something unusual on the amount or type of payment
Type	Integer	date	String	float	string
Value / Range	0-MaxID	date	any	any	any
Null allowed	no	no	no	no	yes
Unique	yes	no	no	no	no
Simple / Composite attribute	simple	simple	simple	simple	single
Single / Multi value	single	multi	single	single	single

Entity: Payment

Description: A payment is given to employee for his days working with the company. Payment can also be given as check, but mostly is cash.

Primary key: payment_ID
Strong/Weak Entity: Strong
Fields to be indexed: payment_ID, amount, date, type, hrs_worked

Attribute name:	payment_ID	date	type	amount	hrs_worked	description
Description	Unique id assigned to each payment to keep track of them	- If cash, date the payment was given. - If check or money order, date on check.	Cash, check,	Amount each employee gets paid	Total hours worked for the company per month	To record if there is something unusual on the amount or type of payment
Type	Integer	date	String	float	float	string
Value / Range	0-MaxID	date	any	any	0.0-MAXID	any
Null allowed	no	no	no	no	no	yes
Unique	yes	no	no	no	no	no
Simple / Composite attribute	simple	simple	simple	simple	simple	single
Single / Multi value	single	multi	single	single	single	single

Entity: Contract

Description: A contract is made for each client's house that the company will start their services with.

Primary key: contract_ID
Strong/Weak Entity: Strong
Fields to be indexed: contract_ID, date, price

Attribute name:	contract_ID	date	price	description
Description	Unique id assigned to each contract to keep track of them	Date the contract was signed	Amount client pays for service	Specification on obligations of landscape company on the client's house.
Type	Integer	date	float	string
Value / Range	0-MaxID	date	any	any
Null allowed	no	no	no	no
Unique	yes	no	no	no
Simple / Composite attribute	simple	simple	simple	single
Single / Multi value	single	multi	single	single

Entity: Tools

Description: this entity will store the tools the company uses to track which tools are taken in the truck in a specific date/route.

Primary key: tools_ID
Strong/Weak Entity: Strong
Fields to be indexed: tools_ID, brand

Attribute name:	**tools_ID**	**brand**	**name**	**description**
Description	Unique id assigned to each tool to keep track of them	Brand of the tool	Name of the tool	Specifications about the tools maintenance
Type	Integer	string	string	string
Value / Range	0-MaxID	any	any	any
Null allowed	no	no	no	yes
Unique	yes	no	no	no
Simple / Composite attribute	simple	simple	simple	single
Single / Multi value	single	single	single	single

Entity: Car

Description: Car is used by the employees to work on the routes, also to store the tools while working.

Primary key: car_ID
Strong/Weak Entity: Strong
Fields to be indexed: car_ID, start_date, end_date

Attribute name:	**car_ID**	**start_date**	**end_date**	**car_desc**
Description	Unique id assigned to each car to keep track of them	Date and hour car was started being in use	Date and hour car ended being in use	Specifications about the car maintenance
Type	Integer	date	date	string
Value / Range	0-MaxID	any	any	any
Null allowed	no	no	no	yes
Unique	yes	no	no	no
Simple / Composite attribute	simple	simple	simple	single
Single / Multi value	single	single	single	single

Entity: House

Description: A house is where the company does its landscape maintenance service every week.

Primary key: house_ID
Strong/Weak Entity: Strong
Fields to be indexed: house_ID, address, start_date, end_date

Attribute name:	**house_ID**	**address**	**start_date**	**end_date**
Description	Unique id assigned to each house to keep track of them	Address of house	Date and hour house service started	Date and hour house service ended
Type	Integer	string	date	date
Value / Range	0-MaxID	any	any	any
Null allowed	no	no	no	no
Unique	yes	no	no	no
Simple / Composite attribute	simple	composite	simple	simple
Single / Multi value	single	single	single	single

Entity: Route

Description: The houses employees have to do service in a specific day are called a Route. There can be different routes in a week. A unique route for every week day.

Primary key: route_ID
Strong/Weak Entity: Strong
Fields to be indexed: route_ID, start_date, end_date

Attribute name:	**route_ID**	**start_date**	**end_date**
Description	Unique id assigned to each route to keep track of them	Date and hour route was started	Date and hour route ended
Type	Integer	date	date
Value / Range	0-MaxID	any	any
Null allowed	no	no	no
Unique	yes	no	no
Simple / Composite attribute	simple	simple	simple
Single / Multi value	single	single	single

Entity: Additional_Service

Description: Additional services given to the client which were not written on the contract or added later, or just one time service.

Primary key: service_ID
Strong/Weak Entity: Strong
Fields to be indexed: service_ID, price

Attribute name:	service_ID	date	price	description
Description	Unique id assigned to each additional service to keep track of them	Date the additional service was done	Amount client paid for additional service	Specification on what done on the service
Type	Integer	date	float	string
Value / Range	0-MaxID	date	any	any
Null allowed	no	no	no	no
Unique	yes	no	no	no
Simple / Composite attribute	simple	simple	simple	single
Single / Multi value	single	multi	single	single

1.2.2 Relationship Type Description

Relationships between two or more entities exists when we associate them with a type. They can be defined by the entities they associate as well as attributes that help describe the relationship. They also distinguish constraints that control the amount of entities related to one another. Relationships can be described by the entities types they associate, the purpose of the relationship, the entity set involved, mapping cardinality, and any additional attributes or constraints that help identify the relationship.

Relationship: produced by
Description: Income is generated from the houses, clients, and services provided to consumers. We describe the relationship between income and contract as income produced by a contract entity.
Entity Sets Involved: Income, Contract
Mapping Cardinality: N .. 1
Descriptive Field: None
Participation Constraint: Income will have total participation while a contract will have partial participation.

Relationship: written for
Description: Every contract will be tied to a house. We describe this relationship between contract and house as contract written for a house.
Entity Sets Involved: Contract, House
Mapping Cardinality: 1 .. 1
Descriptive Field: None
Participation Constraint: A contract will have total participation and house will have total participation since every contract will be tied to a house.

Relationship: provided to
Description: Additional services can be provided to a house that may or may not be under contract. We describe this relationship between additional service and house as an additional service provided to a house.
Entity Sets Involved: Additional Service, House
Mapping Cardinality: N .. 1
Descriptive Field: None
Participation Constraint: An additional service will have partial participation while a house will have total participation if an additional service is being provided.

Relationship: owned by

Description: Every house is owned by someone and in our database we describe these owners as clients. Clients will be the ones being provided and charged for the services. We describe this relationship as a house is owned by a client.

Entity Sets Involved: House, Client

Mapping Cardinality: N .. 1

Descriptive Field: None

Participation Constraint: In this relationship the house has total participation since it depends on a client to own it. Client also is a total participation since every client owns a house.

Relationship: assigned

Description: Every house is assigned a route in order for the employees to keep track of the houses to service during the week. A route will consist of a list or schedule of houses. We describe this relationship as every house is assigned a route.

Entity Sets Involved: House, Route

Mapping Cardinality: 1 .. 1

Descriptive Field: None

Participation Constraint: A house in this relationship will have total participation since every house needs a route to be worked on by a route. A route has partial participation since not every route is tied to a house. A route can have different houses but not assigned to every house in the house set.

Relationship: worked by

Description: A route is worked by an individual car in the set. Every route will be different so we need to distinguish which car is working on which route. We describe this relationship as every route is worked by a car.

Entity Sets Involved: Route, Car

Mapping Cardinality: 1 .. 1

Descriptive Field: None

Participation Constraint: A route will have total participation since every route will have a car it is worked by in the set. A car will have partial participation since if a car changes routes or a car is out of maintenance not every car will have a route.

Relationship: contains

Description: A car will carry all the appropriate tools in order to perform the services required by the employees. A car will be loaded with tools that will be kept track of in order to have an accurate inventory. We describe this relationship as a car contains this set of tools.

Entity Sets Involved: Car, Tool

Mapping Cardinality: 1 .. N

Descriptive Field: None

Participation Constraint: Car in this relationship will have total participation since every car will need a set of tools. Tools will have total participation since every tool will be dependent of the car it is contained in.

Relationship: assign to

Description: A car will car a particular set of employees or an individual employee. The employees will ride in a particular car to go out and perform services required of them. We describe this relationship as a car will be assigned to an employee or set of employees.

Entity Sets Involved: Car, Employee

Mapping Cardinality: 1 .. N

Descriptive Field: Start_Date, End_Date

Participation Constraint: Car will have total participation since every car will be assigned to an employee if in use. Employee will have partial participation because not every employee in the database will necessarily ride in a car but can be assigned a car.

Relationship: paid

Description: Employees that will work for the business have to be paid for the work being provided. In order to keep track of payments and information on those payments to employees we have a payment database. We describe this relationship as an employee is paid a payment.

Entity Sets Involved: Employee, Payment

Mapping Cardinality: 1 .. N

Descriptive Field: None

Participation Constraint: Employee will have partial participation in that an employee is not dependent of a payment. A payment may not be given to an employee if a worker is not being paid for some reason. Payment will have total participation since it will be dependent of an employee and a payment will not be made if no employee existed for it.

1.2.3 Related Entity Type

Specialization is the process of defining subclasses of a specific entity type in which attributes of the superclass are transferred down to the subclass. The subclass then has its own distinguishing attributes that help define it and give reason for the subclass.

Generalization is the reverse process of specialization in that generalization is a bottom-up approach to combining classes into a superclass. and eliminates unnecessary redundancy. Tools in our database is an example of this process, since instead of breaking the tools down into subclasses of different types we generalized it to a tool entity type.

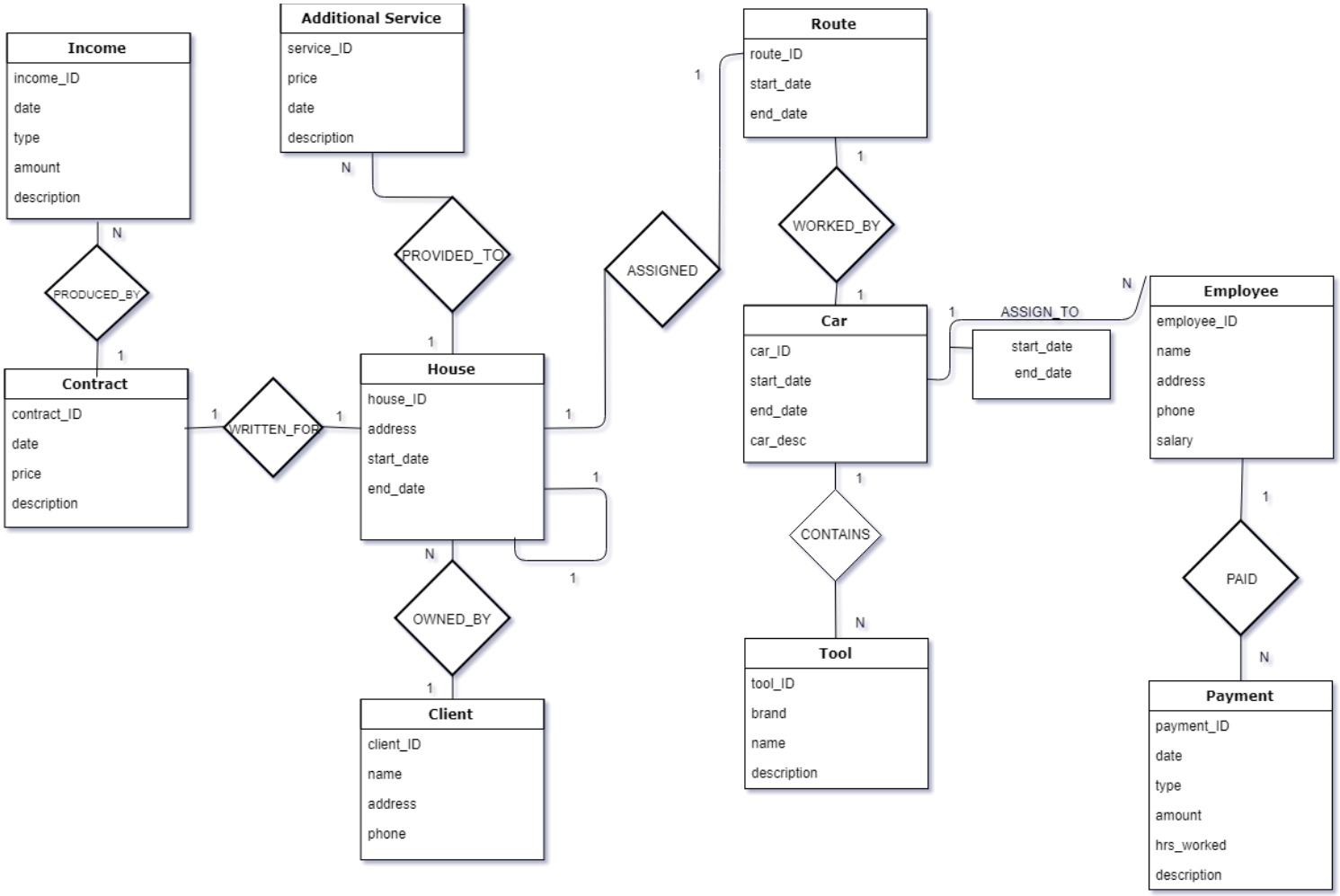
Aggregation is the process of relating two entities as a single entity or group. The purpose is to gather more information by specifying only a certain group or attribute from multiple entities. It creates a whole object instead of individual components.

Specializations and generalizations contain constraints of disjointedness and completeness. The disjointedness constraints specifies that a specialization subclasses are disjoint. This means the entity can only be a only one of the specialization classes. The completeness constraint can either be total or partial. Total completeness says that every entity in a superclass is at least member of one of the subclass. Partial completeness constraint specifies that an entity does not have to be a member of any subclass.

1.2.4 E-R Diagram

A helpful tool to visualize what is being produced using the ER model we make the use of a ER diagram. The ER diagram associates the entities and relationships of our model in a way that can be understood using visual objects and flow. The relationships in the diagram will be represented using lines and connected in between will be a relationship type represented by a diamond box with relationship type name. Entities will be represented using a square box with entity name and attributes written inside the box. Included with the relationships and entities will be symbols “1” or “N” to represent the cardinality of the relationship.

The ER diagram for Green Landscaping is as follows:


Income
income_ID
date
type
amount
description
Additional Service
service_ID
price
date
description
Route
route_ID
start_date
end_date
PRODUCED_BY
PROVIDED_TO
ASSIGNED
WORKED_BY
Contract
contract_ID
date
price
description
WRITTEN_FOR
House
house_ID
address
start_date
end_date
Car
car_ID
start_date
end_date
car_desc
ASSIGN_TO
start_date
end_date
Employee
employee_ID
name
address
phone
salary
CONTAINS
OWNED_BY
PAID
Tool
tool_ID
brand
name
description
Client
client_ID
name
address
phone
Payment
payment_ID
date
type
amount
hrs_worked
description

PHASE 2: Conceptual database design: Using E-R Modeling

2.1 E-R model and Relational model:

- 2.1.1 Description of E-R model and Relational Model
- 2.1.2 Comparison of E-R model and Relational Model

2.2 Conversion from Conceptual Database Model to Logical Database Model

- 2.2.1 Converting Entity Types to Relations
- 2.2.2 Converting Relationship Types to Relations
- 2.2.3 Database Constraints

2.3. Convert Green Landscape Conceptual Database into a Relational Database

- 2.3.1 Relation Schema
- 2.3.2. Sample Data of Relation

2.4. Sample Queries for Database

- 2.4.1 Design Of Queries
- 2.4.2 Relational Algebra Expressions
- 2.4.3 Tuple Relational Calculus Expressions
- 2.4.4 Domain Relational Calculus Expressions