CPCS241-Database I-1st Semester 2023-Project

Party Organization

DB Design

Group No: 3

Student Name	**Student Number**
Rahaf Said Alghamdi	**2006609**
Rolina Fawaz Kattuah	**2006180**
Samar Saad Alotaibi	**2006191**

Contents

PART I: Analysis 4
1 Problem Definition and Data Requirements 4
1.1 Problem Description 4
1.2 Data Requirements 4
1.3 Business Rules 5
1.4 Intended Output of the system 5
PART II: DB DEISGN 6
2 ER Diagram Design 6
2.1 ER diagram 6
2.2 Design of Business Rules 7
3 ER-to-logical schema mapping 8
3.1 Mapping of Regular Entity Types 8
3.2 Mapping of Weak Entity Types 10
3.3 Mapping of binary 1-1 relationship types 11
3.4 Mapping of binary 1-N relationship types 11
3.5 Mapping of binary M-N relationship types 12
3.6 Mapping of multivalued attributes 15
3.7 Mapping of n-ary relationship types 15
3.8 Schema Diagram 16
4 Normalization 17
4.1 First Normal Form 17
4.2 Second Normal Form 19
4.3 Third Normal Form 24
5 Final DB Schema Diagram 28
PART III: IMPLEMENTATION 29
6 Table Creation Script 29
6.1 <*employeeP*> TABLE 29
6.2 <*departmentP*> TABLE 30
6.3 <*addressP*> TABLE 30
6.4 <*clientP*> TABLE 30
6.5 <assistsP > TABLE 31
6.6 <*driverP*> TABLE 31
6.7 <*supplierP*> TABLE 31
6.8 <*partyP*> TABLE 32
6.9 <*providesP*> TABLE 32
6.10 <*choosesP*> TABLE 32

6.11 <*locationP*> TABLE 33
6.12 <*discountP*> TABLE 33
6.13 <*receiptP*> TABLE 33
6.14 <*affectP*> TABLE 34
6.15 <*offersP*> TABLE 34
6.16 <h*asP*> TABLE 34
7 Constraints Script 35
8 Queries 37
8.1 <*List of Supervisees*> 37
8.2 <*Number of Clients*> 37
8.3 <*Yearly Revenue*> 38
8.4 <*Above Average Driver(s)*> 39
8.5 <*Providers of Each Service*> 39
8.6 <*Remaining Days of Coupon*> 40
9 Transactions 41
9.1 <*Salary Raise*> 41
APPENDIX 42

PART I: Analysis

1 Problem Definition and Data Requirements

1.1 Problem Description

Today, it is noticeable that people need to organize their big parties, so in our project we are going to represent a data base for a party organizing company that facilitates planning for clients. The company has several employees who provide the service to the company's clients so that the client specified his services and type of party then the company organizes it for him at the appropriate prices.

1.2 Data Requirements

Employee:
- Has a name (first, middle, last), SSN, ID, salary, gender, phone number, and birthdate.
- Each employee must have a manager.

Client:
- Each client has a name, ID, credit card number, and CNN.
- A client has the cost that he will pay for all parties.
- It's required the client to specify which party type he would like.

Department:
- Each department has a name, address (building number/building floor), and unique number.
- All departments manage several employees.

Party:
- Each party has a unique order ID to identify and modify with.
- Parties have a time and a location.
- Services are offered for each party.

Supplier:
- Each supplier company has a unique trademark name.
- Each company has an e-mail address and a location.
- Every company has multiple drivers.

Discount:
- Discount percentages are used through unique coupons.
- Each discount activated by a coupon has an expiration date.
- Activated discounts have a set number of maximum uses allowed.

Receipt:

Every receipt has a print date and time as well as a unique number.

Driver:
- Each driver has working hours, a name, and an ID used as a partial key.

1.3 Business Rules

Client:
A client must have a valid credit card number and CNN.

Department:
The department may have more than one address.

Party:
Parties’ price is included in the receipt.

Supplier:
Supplier companies ensure the provision of a party’s services for our company.

Discount:
A coupon can be shared between multiple receipts.

Receipt:
- The total cost is calculated through the relationships between the cost and a party’s price/discounts.
- Multiple receipts can simultaneously belong to one client.

Driver:
A driver is identified by his ID and the supplier company he is assigned to.

1.4 Intended Output of the system

Queries:

- Retrieve a list of X manager’s supervisees.
- Display the number of clients each employee is currently assisting in descending order.
- Calculate and display the sum of the company’s yearly revenue.
- Retrieve the information of drivers whose working hours are above the average at the company they are employed in.
- Display the names and contact information of supplier companies currently providing services for specific party types.
- Calculate the remaining days of a coupon’s validity to notify the clients.

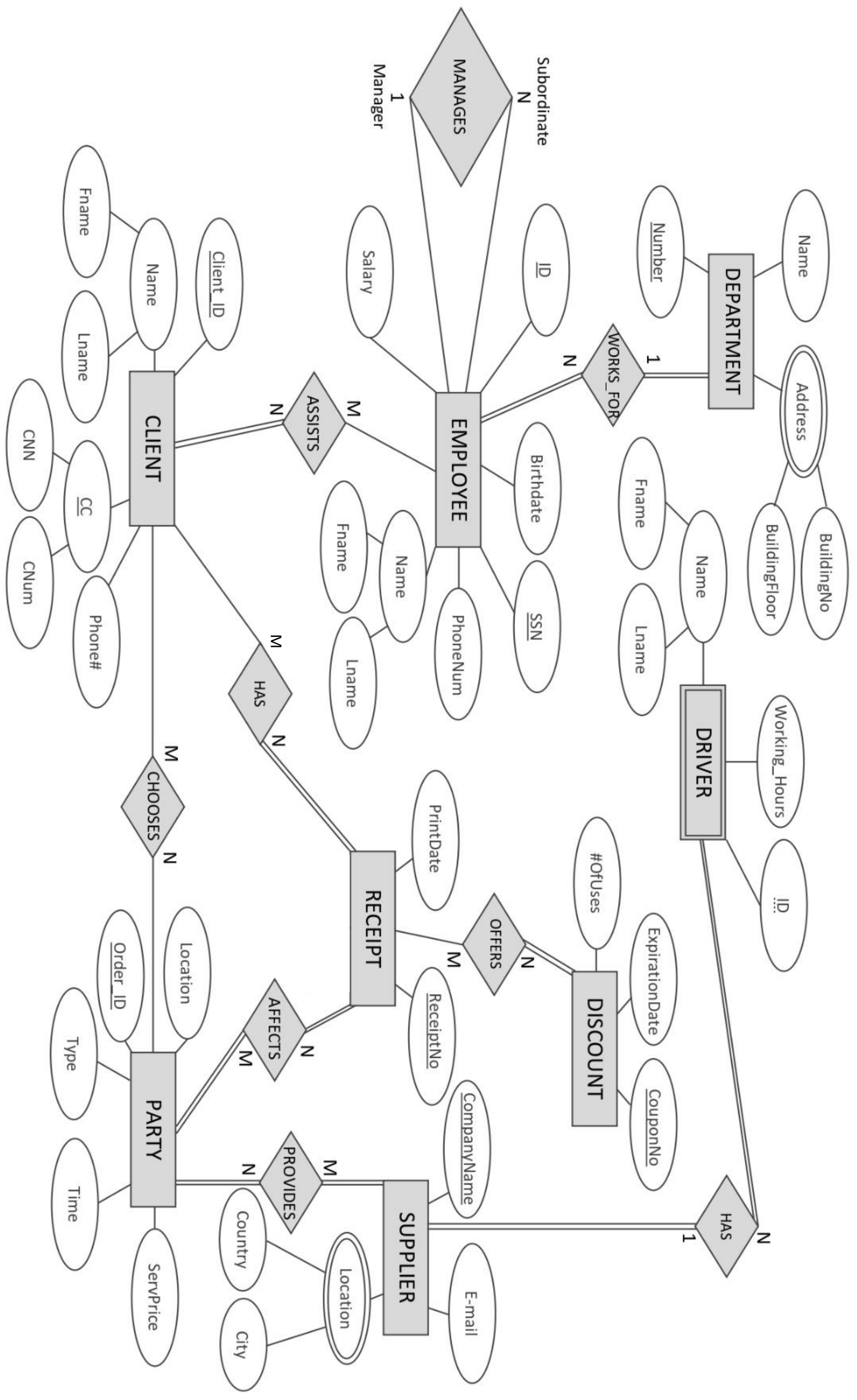
Transactions:

- Update an entity’s information.

PART II: DB DEISGN

2 ER Diagram Design

2.1 ER diagram



2.2 Design of Business Rules

In this subsection, the process in which business rules have been translated into design decisions is shown. Sufficient justification is provided when necessary.

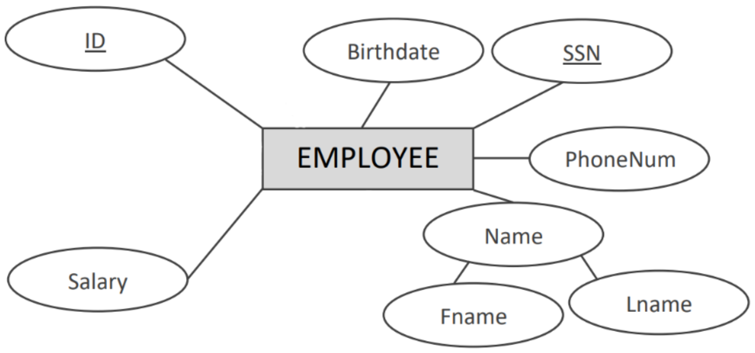
Business Rule	Design Decisions	Justification (if any)
A client must have a valid credit card number and CNN.	A unique attribute.	The credit card number and its CNN cannot be null because the client will use it for paying online.
The department may have more than one address.	A multi-valued attribute.	A department can be in more than one building.
Parties' price is included in the receipt.	M:N relationship between the party and receipt entities.	
Supplier companies ensure the provision of a party's services for our company.	N:M relationship between the party and supplier entities.	For each party, multiple supplier companies can be dealt with.
A coupon can be shared between multiple receipts.	Partial participation of the receipt entity and a M:N relationship between the discount and receipt entity.	Only some clients have coupon codes activated.
The total cost is calculated through the relationships between the cost and discounts.	M:N relationship between the receipt and party entity.	
	M:N relationship between the receipt and discount entity.	
Multiple receipts can simultaneously belong to one client.	M:N relationship between the receipt and client entity.	A client can have multiple orders at a time.
A driver is identified by his ID and the supplier company he is assigned to.	The driver is a weak entity and is dependent on the supplier entity through a N:1 relationship.	Each supplier company has many drivers, each of which work for one company. Drivers' information is stored and retrieved in the supplier companies' database.

3 ER-to-logical schema mapping

3.1 Mapping of Regular Entity Types

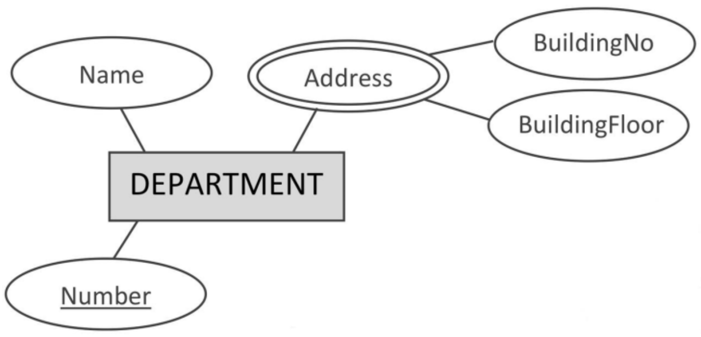
Employee

ID	SSN	Birthdate	PhoneNum	Fname	Lname	Salary



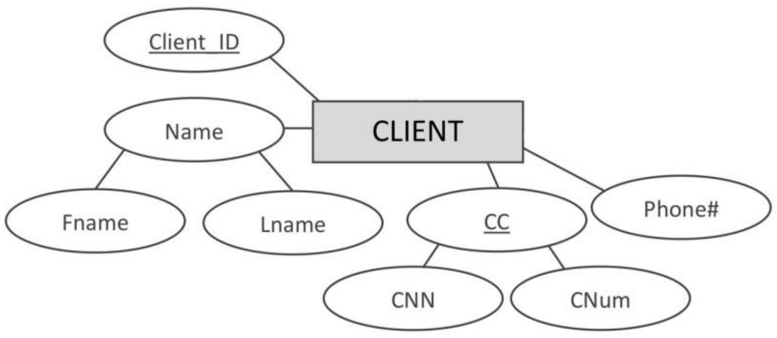
Department

DeptName	DepNum



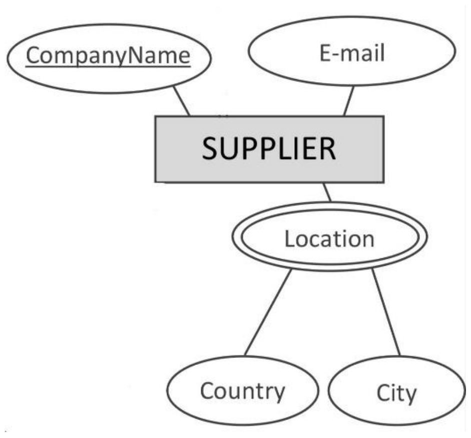
Client

Client_ID	fName	lName	PhoneNum	CreditNum	CardCN



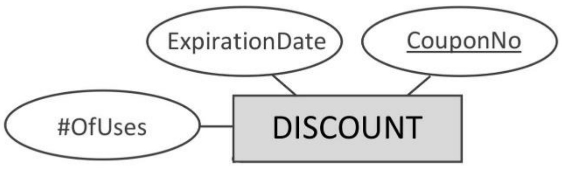
Supplier

CompanyName	E-mail



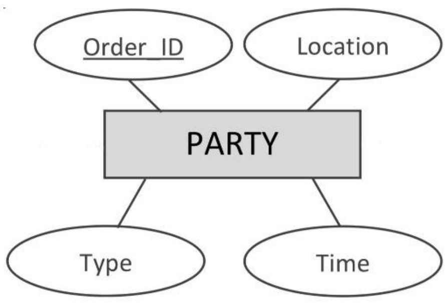
Discount

CouponNo	ExpiraitonDate	#OfUses



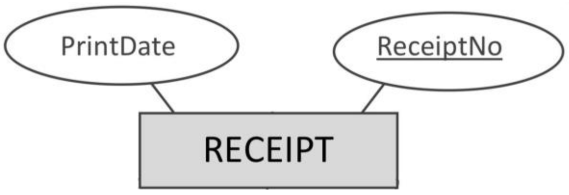
Party

Order_ID	Time	Location	Party_Type



Receipt

ReceiptNum	Print_Date



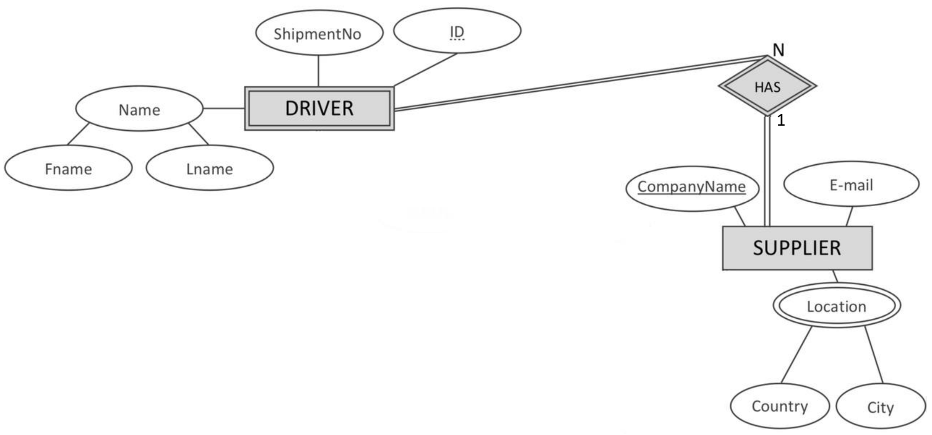
3.2 Mapping of Weak Entity Types

Driver

ID	fName	lName	ShipmentNo	Country	CompName

Supplier

CompanyName	E-mail



3.3 Mapping of binary 1-1 relationship types

There are no 1:1 relationships.

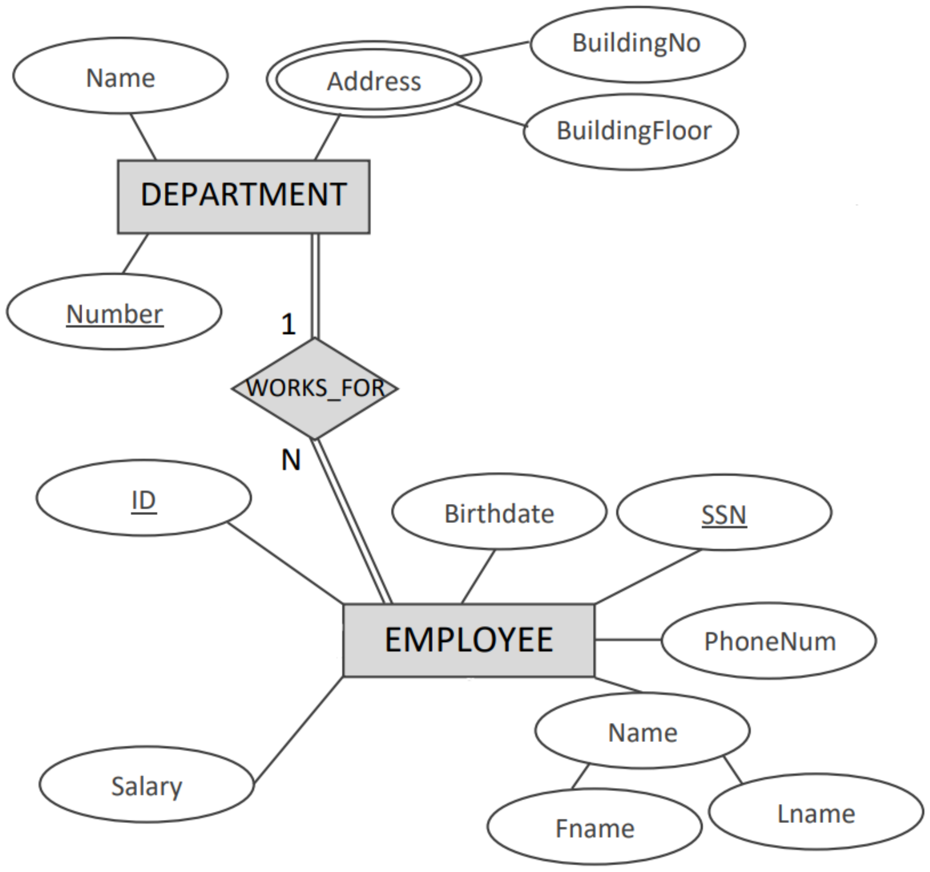
3.4 Mapping of binary 1-N relationship types

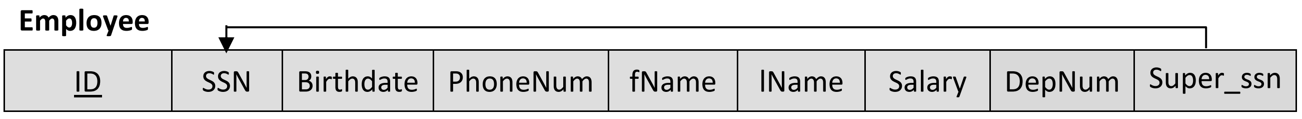
Employee

<u>ID</u>	SSN	Birthdate	PhoneNum	Fname	Lname	Salary	DepNum

Department

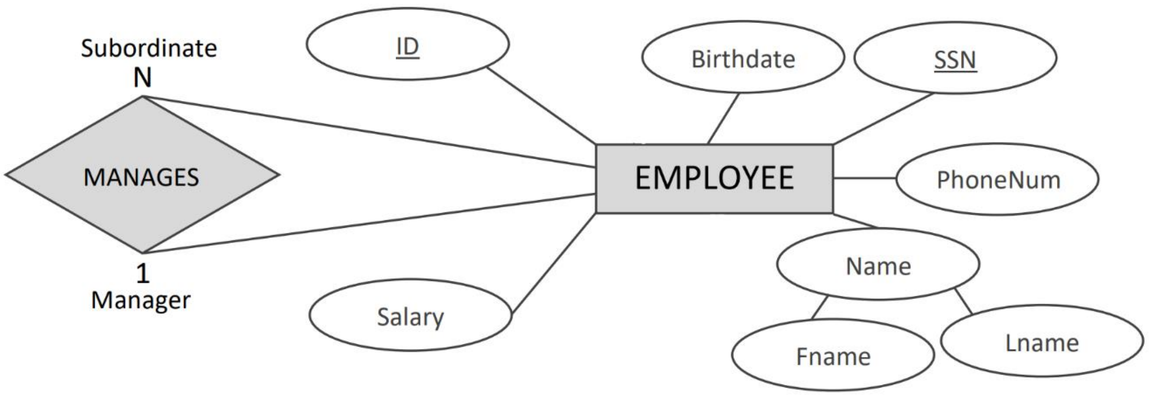
DeptName	<u>DepNum</u>



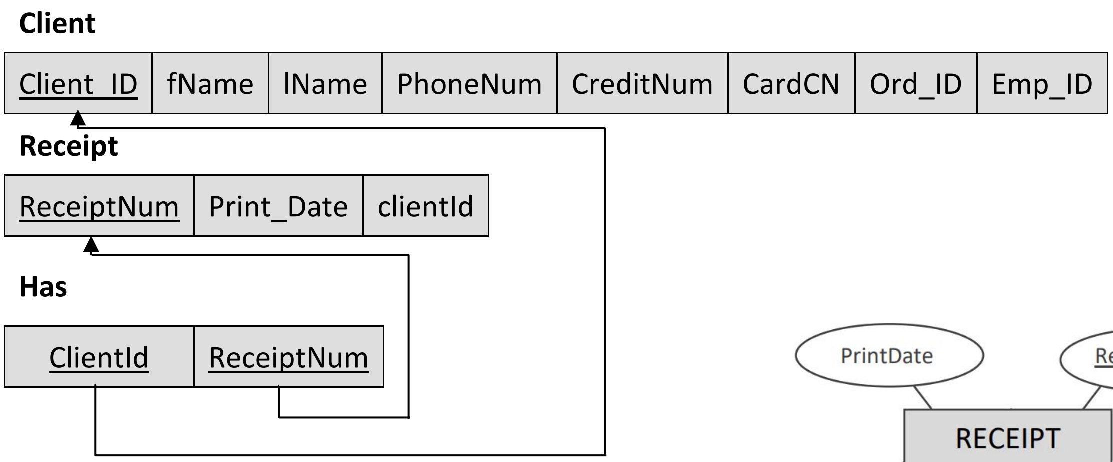


Employee

ID	SSN	Birthdate	PhoneNum	fName	lName	Salary	DepNum	Super_ssn



3.5 Mapping of binary M-N relationship types



Client

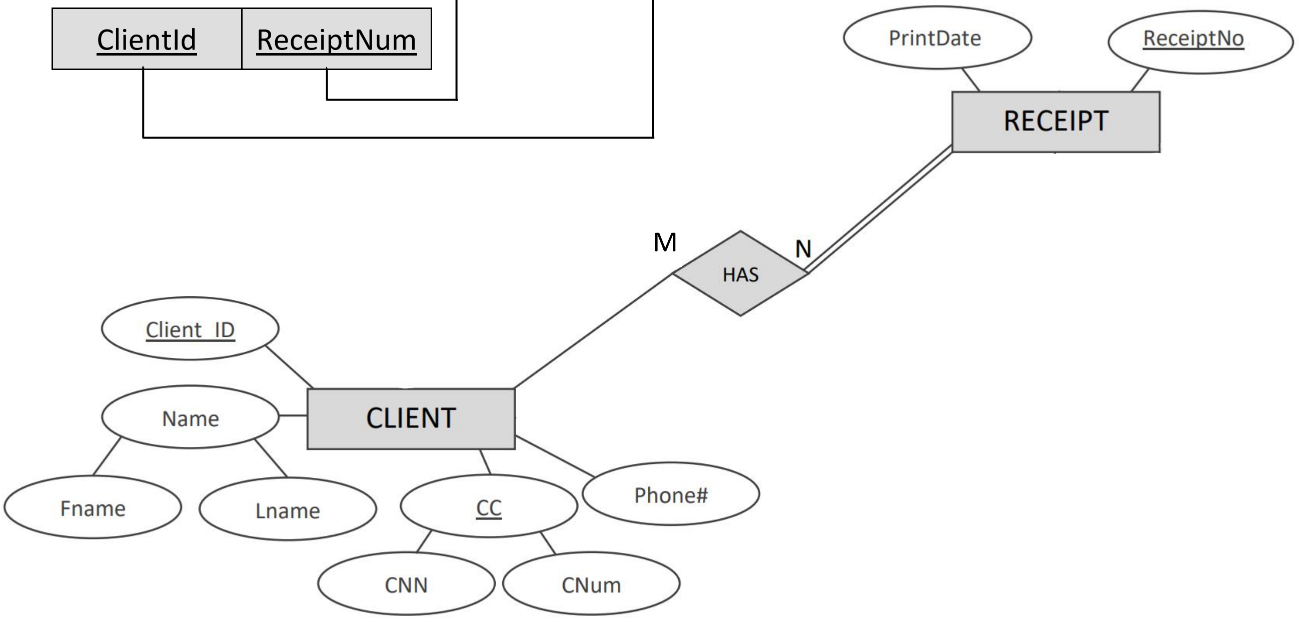
Client_ID	fName	lName	PhoneNum	CreditNum	CardCN	Ord_ID	Emp_ID

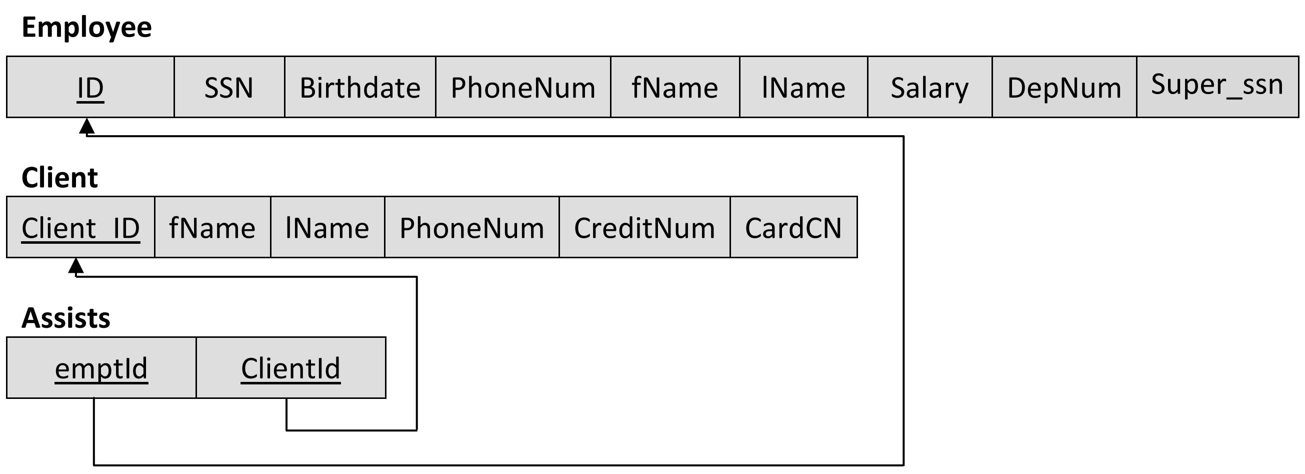
Receipt

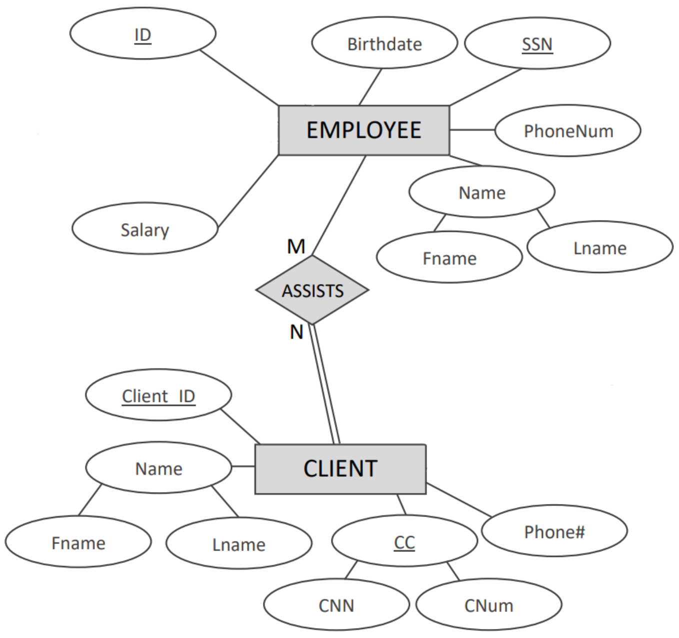
ReceiptNum	Print_Date	clientId

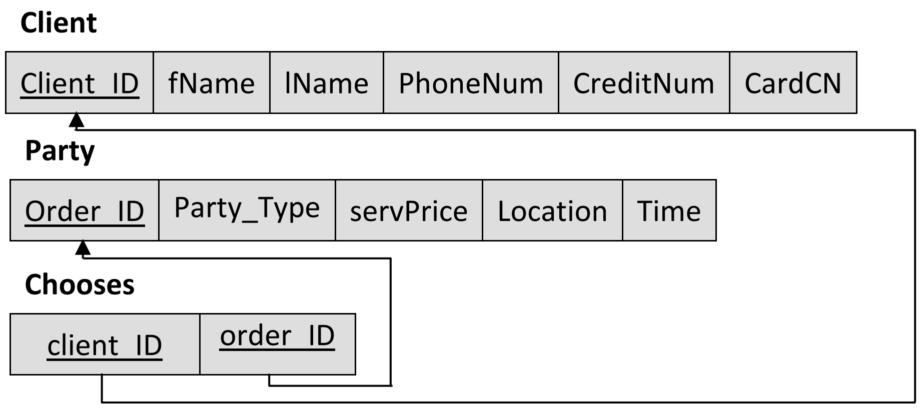
Has

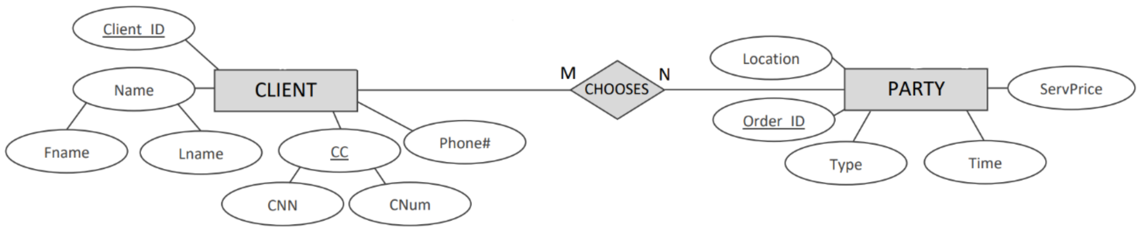
ClientId	ReceiptNum

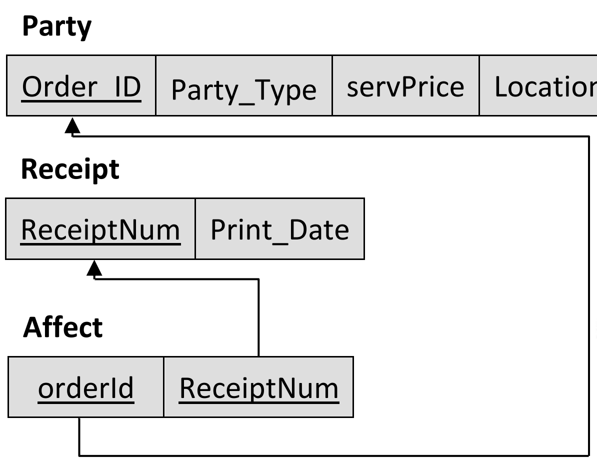



Employee
ID
SSN
Birthdate
PhoneNum
fName
lName
Salary
DepNum
Super_ssn
Client
Client_ID
fName
lName
PhoneNum
CreditNum
CardCN
Assists
emptId
ClientId


ID
Birthdate
SSN
EMPLOYEE
PhoneNum
Name
Salary
Fname
Lname
M
ASSISTS
N
Client_ID
Name
CLIENT
Fname
Lname
CC
Phone#
CNN
CNum


Client
Client_ID
fName
lName
PhoneNum
CreditNum
CardCN
Party
Order_ID
Party_Type
servPrice
Location
Time
Chooses
client_ID
order_ID


Client_ID
Name
CLIENT
Fname
Lname
CC
Phone#
CNN
CNum
M
CHOOSES
N
Location
PARTY
ServPrice
Order_ID
Type
Time



Party

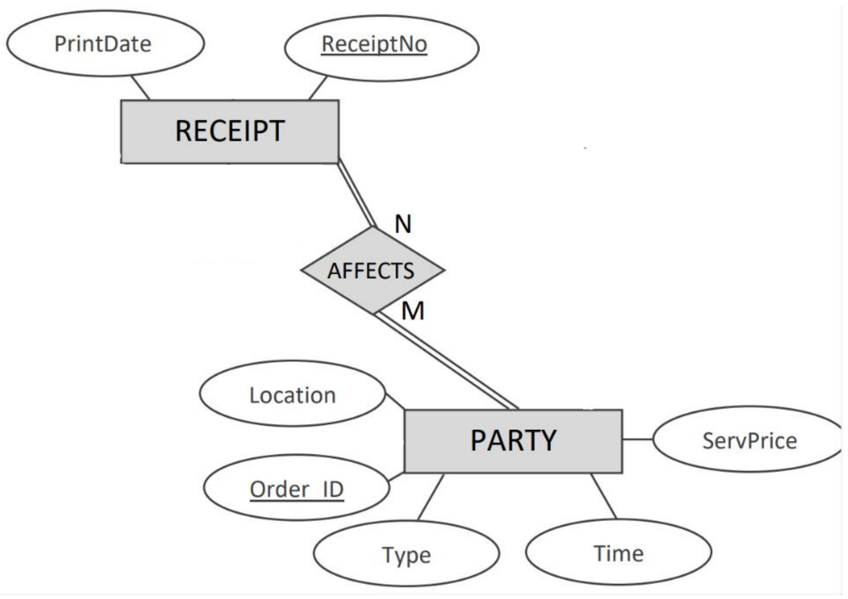
<u>Order_ID</u>	Party_Type	servPrice	Location	Time

Receipt

<u>ReceiptNum</u>	Print_Date

Affect

<u>orderId</u>	<u>ReceiptNum</u>



Supplier

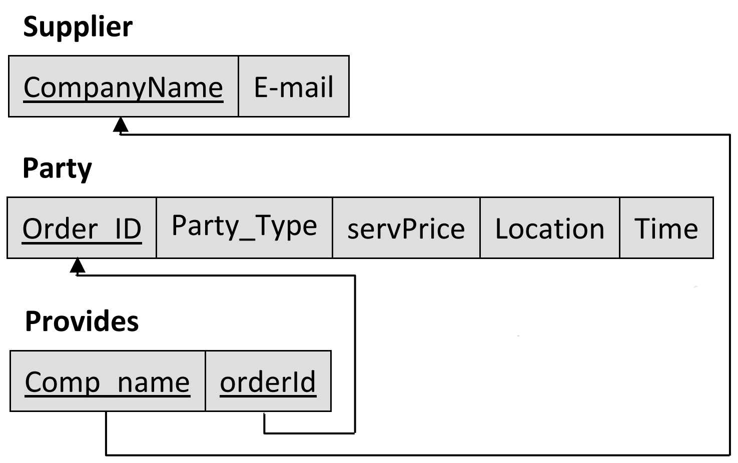
<u>CompanyName</u>	E-mail

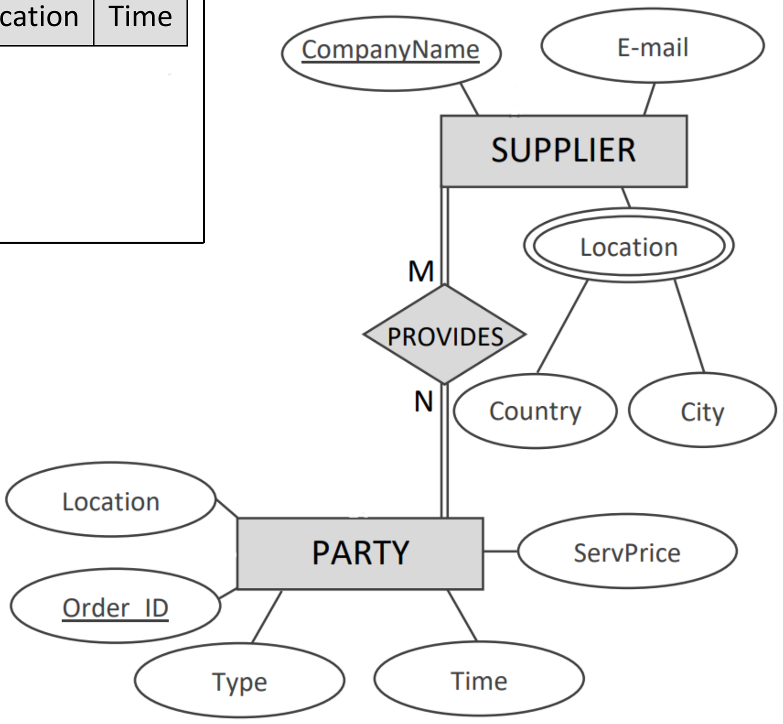
Party

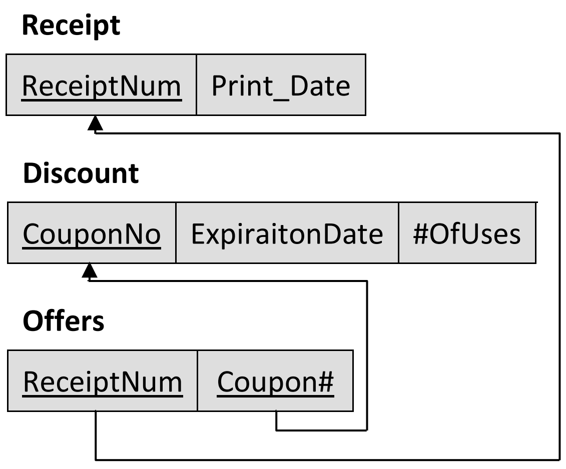
<u>Order_ID</u>	Party_Type	servPrice	Location	Time

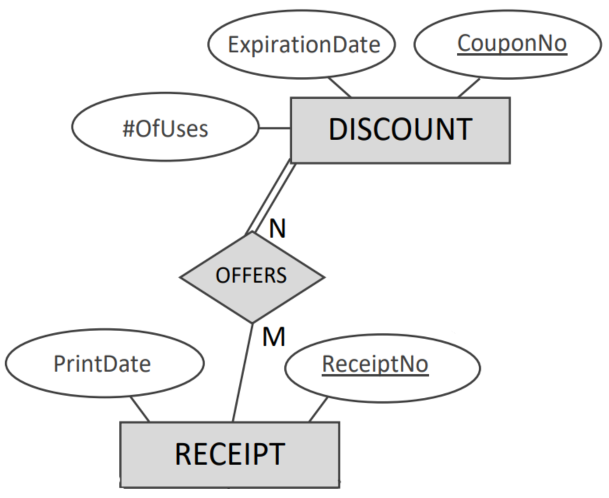
Provides

<u>Comp_name</u>	<u>orderId</u>

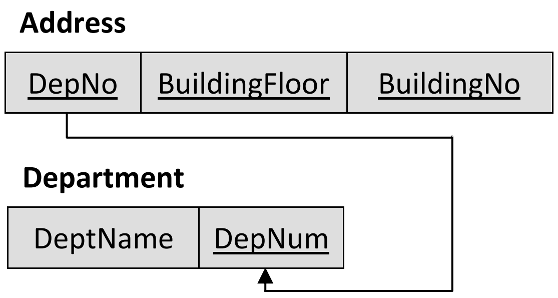


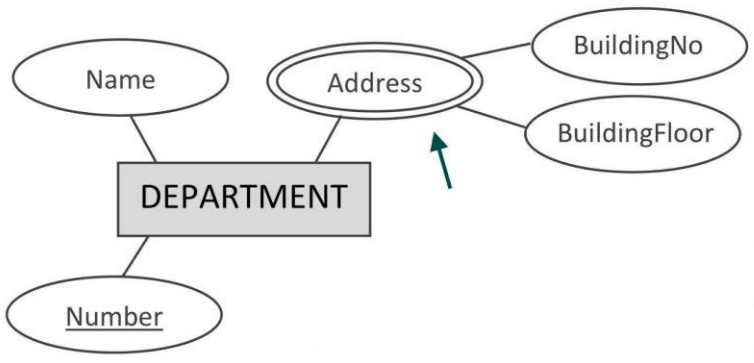


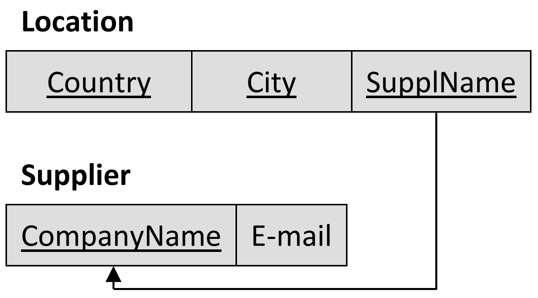


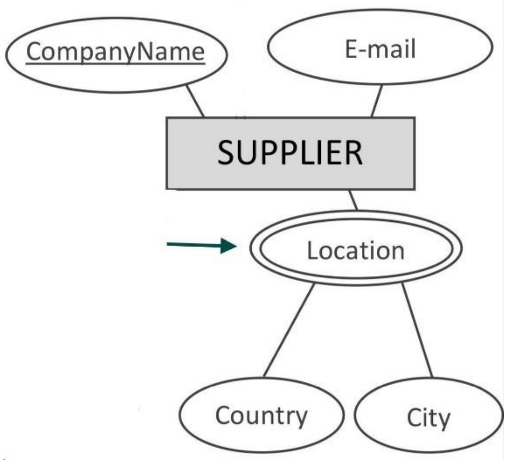


3.6 Mapping of multivalued attributes





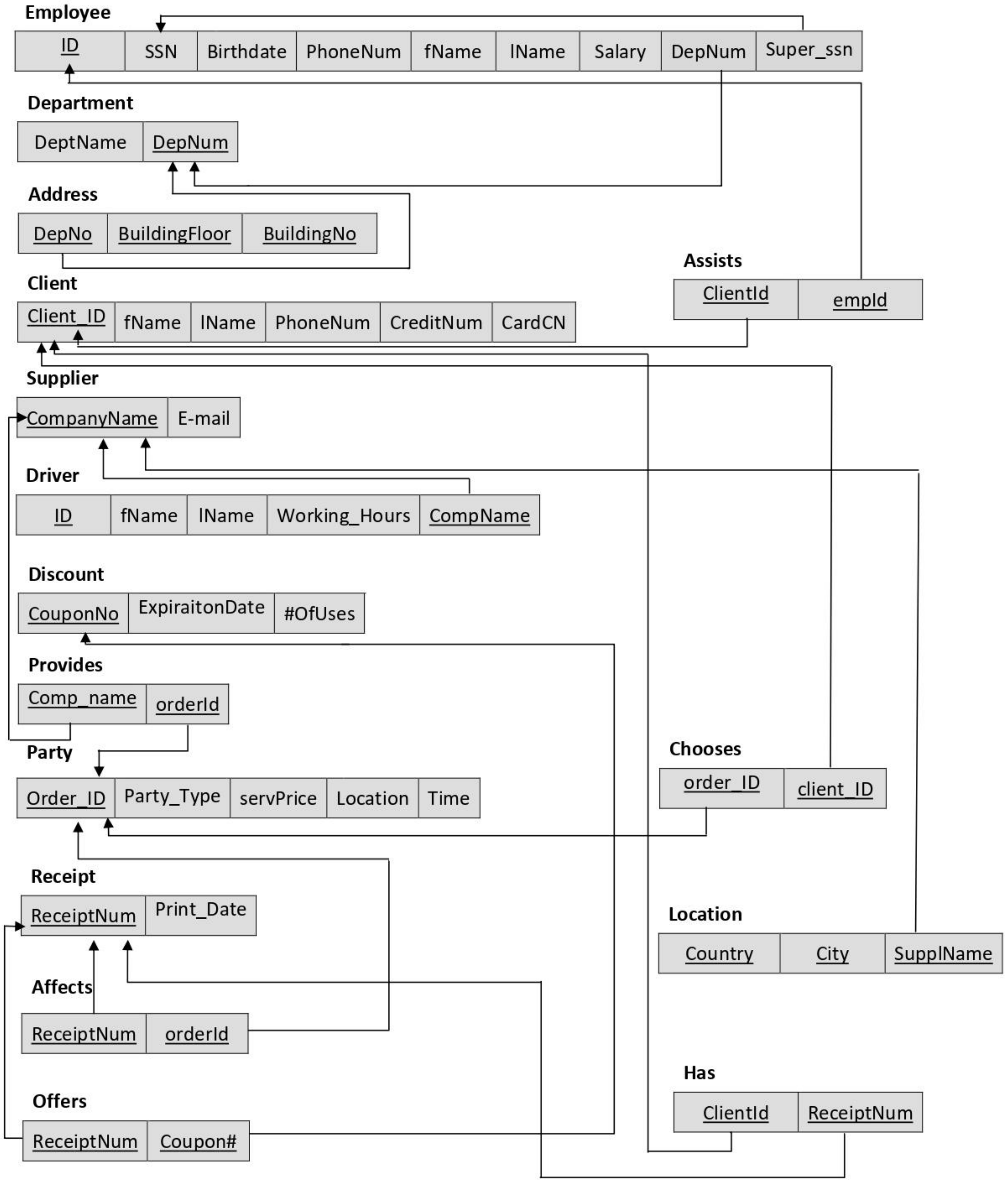




3.7 Mapping of n-ary relationship types

There are no n-ary relationships.

3.8 Schema Diagram



4 Normalization

4.1 First Normal Form

For a relation schema R to be in the first normalization form (1NF), it's requires not having multivalued and composite attribute. We transformed the multivalued attributes (Address and location) to a separate relation, for that we don't have anything that goes against for first normalization form.

Employee

ID	SSN	Birthdate	PhoneNum	fName	lName	Salary	DepNum	Super_ssn

Department

DeptName	DepNum

Client

Client_ID	fName	lName	PhoneNum	CreditNum	CardCN

Supplier

CompanyName	E-mail

Driver

ID	fName	lName	Working_Hours	CompName

Discount

CouponNo	ExpiraitonDate	#OfUses

Party

Order_ID	Party_Type	servPrice	Location	Time

Receipt

ReceiptNum	Print_Date

- **Address**
  It's a multivalued attribute and we transformed it to a relation R.

Address

DepNo	BuildingFloor	BuildingNo

- **Location**
  It's a multivalued attribute and we transformed it to a relation R.

Location

Country	City	SupplName

Provides

Comp_name	orderId

Affects

ReceiptNum	orderId

Offers

ReceiptNum	Coupon#

Assists

ClientId	empId

Chooses

order_ID	client_ID

Has

ClientId	ReceiptNum

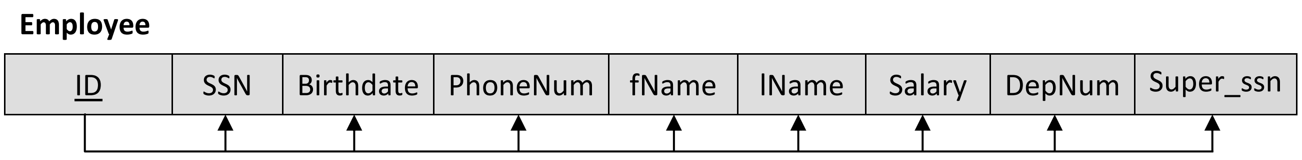
4.2 Second Normal Form

For a relation schema R to be in second normal form (2NF), every non-prime attribute A in R must be fully functionally dependent on the primary key.

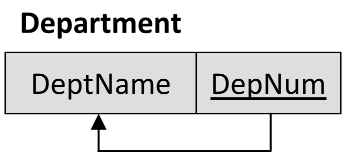
A prime attribute is an attribute that is a member of the primary key K, whereas a non-prime attribute is one that contradicts the above definition.

The process of normalizing the relation R in 2NF is achieved via decomposing it into separate relations where each attribute is grouped with the PK is it dependent on

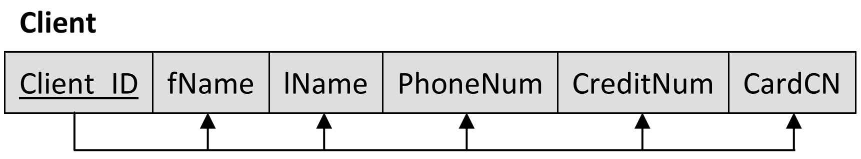
- **Employee**
  ID is the only prime attribute of the Employee relation. All non-prime attributes are fully functionally dependent on ID; therefore, no alteration is made.



- **Department**
  DepNum is the only prime attribute of the Department relation. All non-prime attributes are fully functionally dependent on DepNum; therefore, no alteration is made.

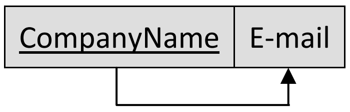


- **Client**
  Client_ID is the only prime attribute of the Client relation. All non-prime attributes are fully functionally dependent on Client_ID; therefore, no alteration is made.



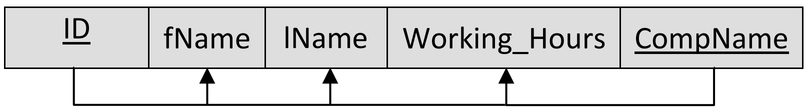
- **Supplier**
  CompanyName is the only prime attribute of the Supplier relation. All non-prime attributes are fully functionally dependent on CompanyName; therefore, no alteration is made.

Supplier



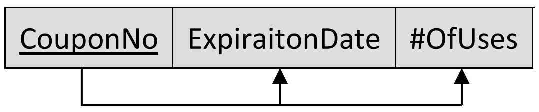
- **Driver**
  ID and CompName are the only prime attributes of the Driver relation. All non-prime attributes of the relation are fully functionally dependent on the whole key; therefore, no alteration is made.

Driver



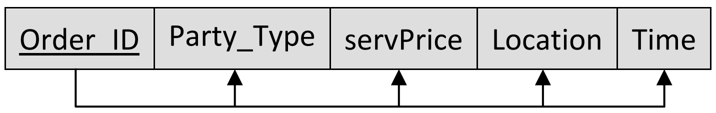
- **Discount**
  CouponNo is the only prime attribute of the Discount relation. All non-prime attributes are fully functionally dependent on CouponNo; therefore, no alteration is made.

Discount



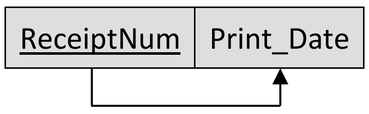
- **Party**
  Order_ID is the only prime attribute of the Party relation. All non-prime attributes are fully functionally dependent on Order_ID; therefore, no alteration is made.

Party



- **Receipt**
  ReceiptNum is the only prime attribute of the Receipt relation. All non-prime attributes are fully functionally dependent on ReceiptNum; therefore, no alteration is made.

Receipt



- **Address**
  DepNo, BuildingFloor, and BuildingNo are the only prime attributes of the Address relation. The relation contains no non-prime attributes to check; therefore, no alteration is made.

Address

DepNo	BuildingFloor	BuildingNo

- **Location**
  Country, City, and SupplName are the only prime attributes of the Location relation. The relation contains no non-prime attributes to check as it is a representation of a multi-valued attribute; therefore, no alteration is made.

Location

Country	City	SupplName

- **Provides**
  Comp_name and orderId are the only prime attributes of the Provides relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Provides

Comp_name	orderId

- **Affects**
  ReceiptNum and orderId are the only prime attributes of the Affects relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Affects

ReceiptNum	orderId

- **Offers**
  ReceiptNum and Coupon# are the only prime attributes of the Offers relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Offers

ReceiptNum	Coupon#

- **Assists**
  ClientId and empId are the only prime attributes of the Assists relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Assists

ClientId	empId

- **Chooses**
  order_ID and client_ID are the only prime attributes of the Chooses relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Chooses

order_ID	client_ID

- **Has**
  ClientId and ReceiptNum are the only prime attributes of the Has relation. The relation contains no non-prime attributes to check due to its M:N cardinality; therefore, no alteration is made.

Has

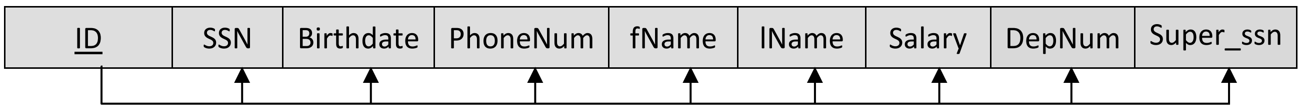
ClientId	ReceiptNum

4.3 Third Normal Form

A Third Normal Form (3NF) relation should not contain a non-primary attribute that transitively depends on the primary key of the relation. All attributes should depend only on the primary key directly.

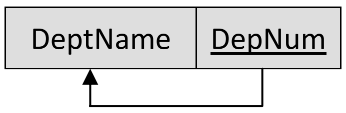
- **Employee**
  Employee relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.

Employee



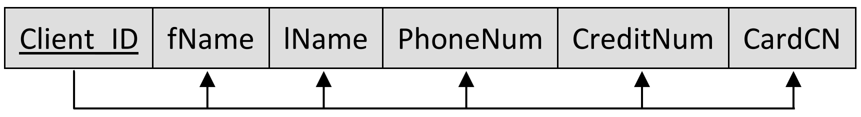
- **Department**
  Department relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.

Department

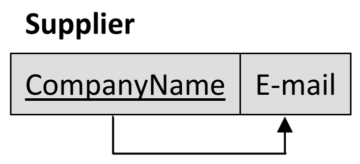


- **Client**
  Client relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.

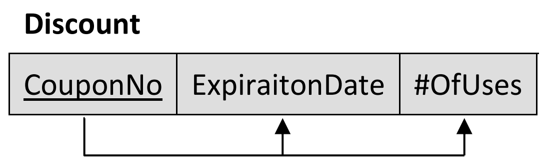
Client



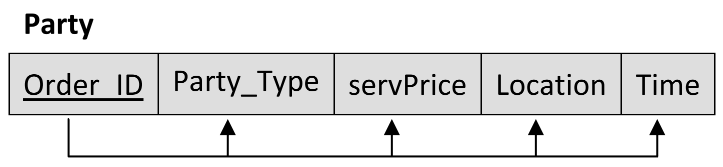
- **Supplier**
  Supplier relation is already in 3NF, since E-mail attribute depends on the key, and there are no transitive dependencies.



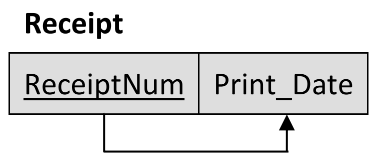
- **Discount**
  Discount relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.



- **Party**
  Party relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.

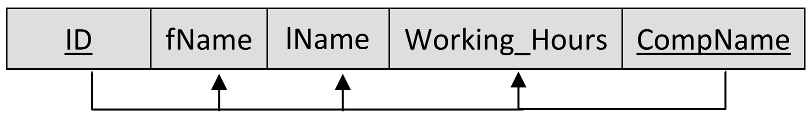


- **Receipt**
  Receipt relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.



- **Driver**
  Driver relation is already in 3NF, since all attributes depends on the key, and there are no transitive dependencies.

Driver



- **Provides**
  The relation has no non-primary keys to be checked (no regular attributes).

Provides

<u>Comp_name</u>	<u>orderId</u>

- **Affects**
  The relation has no non-primary keys to be checked (no regular attributes).

Affects

<u>ReceiptNum</u>	<u>orderId</u>

- **Offers**
  The relation has no non-primary keys to be checked (no regular attributes).

Offers

<u>ReceiptNum</u>	<u>Coupon#</u>

- **Assists**
  The relation has no non-primary keys to be checked (no regular attributes).

Assists

<u>ClientId</u>	<u>empId</u>

- **Chooses**
  The relation has no non-primary keys to be checked (no regular attributes).

Chooses

<u>order_ID</u>	<u>client_ID</u>

- **Has**
  The relation has no non-primary keys to be checked (no regular attributes).

Has

<u>ClientId</u>	<u>ReceiptNum</u>

- **Address**
  The relation has no non-primary keys to be checked (no regular attributes).

Address

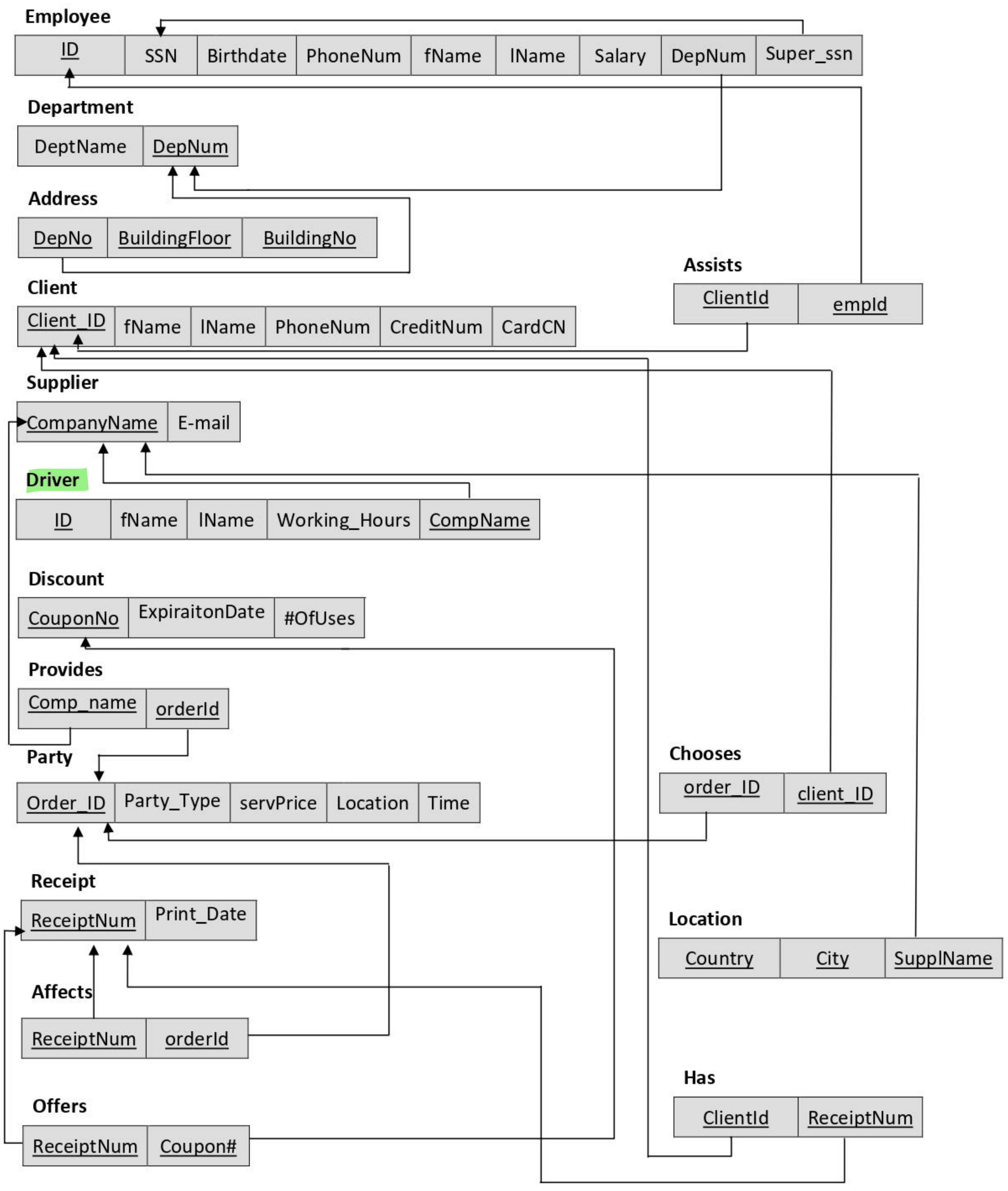
<u>DepNo</u>	<u>BuildingFloor</u>	<u>BuildingNo</u>

- **Location**
  The relation has no non-primary keys to be checked (no regular attributes).

Location

<u>Country</u>	<u>City</u>	<u>SupplName</u>

5 Final DB Schema Diagram



PART III: IMPLEMENTATION

6 Table Creation Script

6.1 <*employeeP*> TABLE

```
create table employeeP(

ID number(10) not null,

ssn number(7) not null,

birthDate date,

phoneNum number(10),

fName varchar2(10),

lName varchar2(20),

salary number(6),

depNum number(3),

superSSN number(7),

unique(ssn),

constraint primaryEmployee primary key(ID));


alter table employeeP

   add foreign key (superSSN) references employeeP(ssn) on delete cascade;


alter table employeeP

   add foreign key(depNum) references departmentP(depNum) on delete cascade;
```

6.2 <*departmentP*> TABLE

```
create table departmentP(
   depName varchar2(20),
   depNum number(3) not null,
   constraint primaryDepartment primary key(depNum));
```

6.3 <*addressP*> TABLE

```
create table addressP(
   depNo number(3) not null,
   bFloor number(3) not null,
   bNo number(3) not null,
   constraint primaryAddressP primary key(depNo,bFloor,bNo),
   foreign key(depNo) references departmentP(depNum) on delete cascade);
```

6.4 <*clientP*> TABLE

```
 create table clientP(
    clientID number(10) not null,
    fName varchar2(10),
    lName varchar2(20),
    phoneNum number(10),
    creditNum number(16) not null,
    cardCN number(3) not null,
    constraint primaryClientP primary key(clientID),
    unique(creditNum,cardCN));
```

6.5 <assistsP > TABLE

```
create table assistsP(
    clientID number(10) not null,
    empID number(10) not null,
    constraint primaryAssists primary key(clientID,empID),
    foreign key(clientID) references clientP(clientID) on delete cascade,
    foreign key(empID) references employeeP(ID) on delete cascade);
```

6.6 <*driverP*> TABLE

```
create table driverP(
    ID number(10) not null,
    compName varchar2(20) not null,
    fName varchar2(20),
    lName varchar2(20),
    workingHours float,
    constraint primaryDriverS primary key(ID,compName));
```

6.7 <*supplierP*> TABLE

```
create table supplierP(
    companyName varchar2(20) not null,
    email varchar2(50),
    constraint primarySupplierP primary key(companyName));
```

6.8 <*partyP*> TABLE

```
create table partyP(
    orderID number(4) not null,
    partyType varchar2(30),
    servPrice number(6,3),
    location varchar2(100),
    partyTime date,
    constraint primaryPartyP primary key (orderID));
```

6.9 <*providesP*> TABLE

```
create table providesP(
    compName varchar2(20) not null,
    orderID number(4) not null,
    constraint primaryProvidesP primary key(compName,orderID),
    foreign key(compName) references supplierP(companyName)on delete cascade,
    foreign key(orderID) references partyP(orderID) on delete cascade);
```

6.10 <*choosesP*> TABLE

```
create table choosesP(
    orderID number(4) not null,
    clientID number(10) not null,
    constraint primaryChooses primary key(orderID,clientID),
    foreign key(orderID) references partyP(orderID) on delete cascade,
    foreign key(clientID)references clientP(clientID)on delete cascade);
```

6.11 <*locationP*> TABLE

```
create table locationP(
    country varchar2(30) not null,
    city varchar2(30) not null,
    supplName varchar2(2) not null,
    constraint primaryLocationP primary key(country,city,supplName),
    foreign key(supplName) references supplierP(companyName) on delete cascade);
```

6.12 <*discountP*> TABLE

```
create table discountP(
    couponNo varchar2(6) not null,
    expirationDate date,
    numOfUses number(2) check(numOfUses<=5),
    constraint primaryDiscountP primary key(couponNo));
```

6.13 <*receiptP*> TABLE

```
create table receiptP(
    receiptNum number(5) not null,
    printTime timestamp,
    constraint primaryReceiptP primary key(receiptNum));
```

6.14 <*affectP*> TABLE

```
create table affectP(
    receiptNum number(5) not null,
    orderID number(4) not null,
    constraint primaryAffectP primary key(receiptNum, orderID),
    foreign key(receiptNum) references receiptP (receiptNum) on delete cascade,
    foreign key(orderID) references partyP(orderID) on delete cascade);
```

6.15 <*offersP*> TABLE

```
create table offersP(
    receiptNum number(5) not null,
    couponNum varchar2(6) not null,
    constraint primaryOffersP primary key(receiptNum, couponNum),
    foreign key(receiptNum) references receiptP (receiptNum) on delete cascade,
    foreign key(couponNum) references discountP (couponNo) on delete cascade);
```

6.16 <*hasP*> TABLE

```
create table hasP(
    clientID number(10) not null,
    receiptNum number(5) not null,
    constraint primaryHasP primary key(clientID, receiptNum),
    foreign key(clientID) references clientP (clientID) on delete cascade,
    foreign key(receiptNum) references receiptP (receiptNum) on delete cascade);
```

7 Constraints Script

In this subsection, the way in which the business rules have been translated into SQL script is shown.

Business Rule	SQL Script	Table
A client must have a valid credit card number and CNN.	creditNum number(16) not null, cardCN number(3) not null, unique(creditNum,cardCN));	Client
The department may have more than one address.	create table addressP(depNo number(3) not null, bFloor number(3) not null, bNo number(3) not null, constraint primaryAddressP primary key(depNo,bFloor,bNo), foreign key(depNo) references departmentP(depNum) on delete cascade);	Address
Parties' price is included in the receipt.	create table affectP(receiptNum number(5) not null, orderID number(4) not null, constraint primaryAffectP primary key(receiptNum, orderID), foreign key(receiptNum) references receiptP (receiptNum) on delete cascade, foreign key(orderID) references partyP(orderID) on delete cascade);	Affect
Supplier companies ensure the provision of a party's services for our company.	create table providesP(compName varchar2(20) not null, orderID number(4) not null, constraint primaryProvidesP primary key(compName,orderID), foreign key(compName) references supplierP(companyName)on delete cascade, foreign key(orderID) references partyP(orderID) on delete cascade);	Provides
A coupon can be shared between multiple receipts.	create table offersP(receiptNum number(5) not null, couponNum varchar2(6) not null,	Offers

	constraint primaryOffersP primary key(receiptNum, couponNum), foreign key(receiptNum) references receiptP (receiptNum) on delete cascade, foreign key(couponNum) references discountP (couponNo) on delete cascade);	
The total cost is calculated through the relationships between the cost and discounts.	create table affectP(receiptNum number(5) not null, orderID number(4) not null, constraint primaryAffectP primary key(receiptNum, orderID), foreign key(receiptNum) references receiptP (receiptNum) on delete cascade, foreign key(orderID) references partyP(orderID) on delete cascade);	Affect
	create table offersP(receiptNum number(5) not null, couponNum varchar2(6) not null, constraint primaryOffersP primary key(receiptNum, couponNum), foreign key(receiptNum) references receiptP (receiptNum) on delete cascade, foreign key(couponNum) references discountP (couponNo) on delete cascade);	Offers
Multiple receipts can simultaneously belong to one client.	create table hasP(clientID number(10) not null, receiptNum number(5) not null, constraint primaryHasP primary key(clientID, receiptNum), foreign key(clientID) references clientP (clientID) on delete cascade, foreign key(receiptNum) references receiptP (receiptNum) on delete cascade);	Has
A driver is identified by his ID and the supplier company he is assigned to.	create table driverP(ID number(10) not null, compName varchar2(20) not null, fName varchar2(20), lName varchar2(20), workingHours float, constraint primaryDriverS primary key(ID,compName));	Driver

8 Queries

In the following subsections, SQL queries which implement the indented output of the database's system (section 1.4) is written down.

8.1 *<List of Supervisees>*

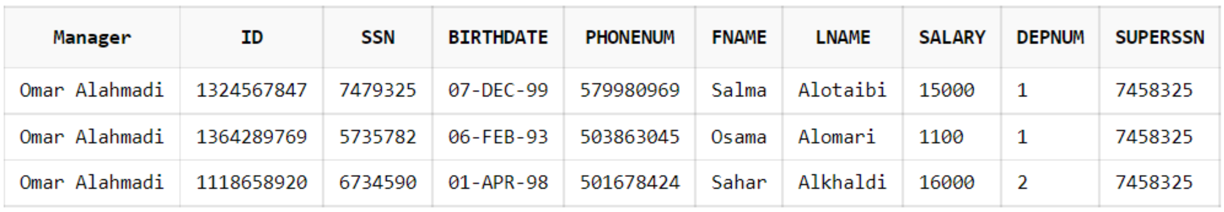
Query in natural language (ENGLISH)

Retrieve a list of X manager's supervisees. Manager with SSN of 7458325 is chosen for demonstration.

SQL script

```
SELECT A0.fName || ' ' || A0.lName AS "Manager", B0.*

FROM employeeP A0, employeeP B0

WHERE A0.SSN = 7458325 AND B0.superSSN = 7458325;
```

Caption of the first five rows of the output



Manager	ID	SSN	BIRTHDATE	PHONENUM	FNAME	LNAME	SALARY	DEPNUM	SUPERSSN
Omar Alahmadi	1324567847	7479325	07-DEC-99	579980969	Salma	Alotaibi	15000	1	7458325
Omar Alahmadi	1364289769	5735782	06-FEB-93	503863045	Osama	Alomari	1100	1	7458325
Omar Alahmadi	1118658920	6734590	01-APR-98	501678424	Sahar	Alkhaldi	16000	2	7458325

8.2 *<Number of Clients>*

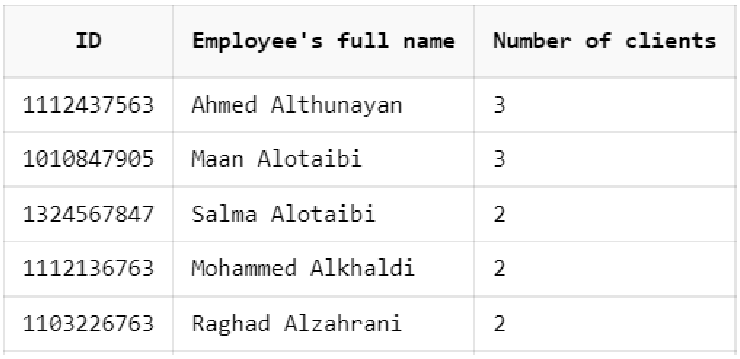
Query in natural language (ENGLISH)

Display the number of clients each employee is currently assisting in descending order.

SQL script

```
SELECT ID,

    A.fName || ' ' || A.lName AS "Employee's full name",

    COUNT(B.clientID) AS "Number of clients"

FROM employeeP A, assistsP B, clientP C

WHERE ID = empID AND C.clientID = B.clientID

GROUP BY ID, A.fName, A.lName

ORDER BY count(B.clientID) DESC;
```

<u>Caption of the first five rows of the output</u>



ID	Employee's full name	Number of clients
1112437563	Ahmed Althunayan	3
1010847905	Maan Alotaibi	3
1324567847	Salma Alotaibi	2
1112136763	Mohammed Alkhaldi	2
1103226763	Raghad Alzahrani	2

8.3 *<Yearly Revenue>*

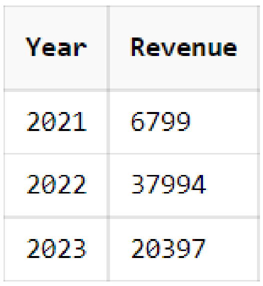
<u>Query in natural language (ENGLISH)</u>

Calculate and display the sum of the company’s yearly revenue.

<u>SQL script</u>

```
SELECT EXTRACT(YEAR FROM partyTime) AS "Year", SUM(servPrice) AS "Revenue"
FROM partyP
GROUP BY EXTRACT(YEAR FROM partyTime);
```

<u>Caption of the first five rows of the output</u>



Year	Revenue
2021	6799
2022	37994
2023	20397

8.4 *<Above Average Driver(s)>*

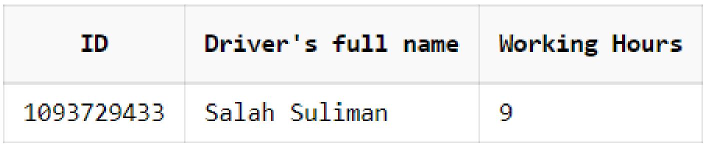
Query in natural language (ENGLISH)

Retrieve the information of drivers whose working hours are above the average at the company they are employed in.

SQL script

```
SELECT ID, fName || ' ' || lName AS "Driver's full name", workingHours AS "Working Hours"
FROM driverP
WHERE compName = 'Amazon'
GROUP BY ID, fName, lName, workingHours
HAVING workingHours > (SELECT avg(workingHours) FROM driverP WHERE compName =
'Amazon');
```

Caption of the first five rows of the output



ID	Driver's full name	Working Hours
1093729433	Salah Suliman	9

8.5 *<Providers of Each Service>*

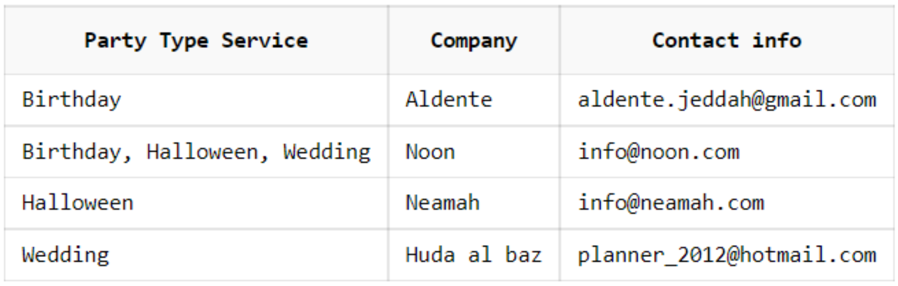
Query in natural language (ENGLISH)

Display the names and contact information of supplier companies currently providing services for specific party types.

SQL script

```
SELECT partyType AS "Party Type Service",
           compName AS "Company",
           email AS "Contact info"
FROM providesP A, partyP B, supplierP C
WHERE compName NOT IN 'Noon'
    AND A.orderID = B.orderID
    AND A.compName = C.companyName

UNION

SELECT (SELECT LISTAGG(DISTINCT B.partyType, ', ') WITHIN GROUP (ORDER BY
           A.compName) FROM providesP A, partyP B, supplierP C),
           A.compName,
           C.email
FROM providesP A, partyP B, supplierP C
WHERE compName IN 'Noon'
    AND A.orderID = B.orderID
    AND A.compName = C.companyName
    GROUP BY B.partyType, A.compName, C.email;
```

Caption of the first five rows of the output



Party Type Service	Company	Contact info
Birthday	Aldente	aldente.jeddah@gmail.com
Birthday, Halloween, Wedding	Noon	info@noon.com
Halloween	Neamah	info@neamah.com
Wedding	Huda al baz	planner_2012@hotmail.com

8.6 <Remaining Days of Coupon>

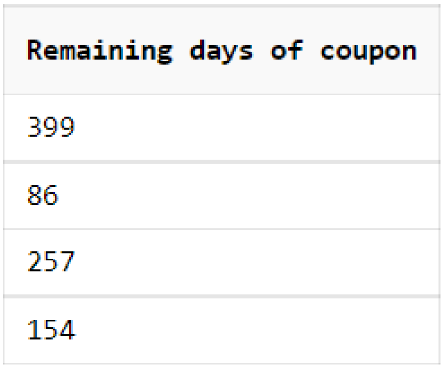
Query in natural language (ENGLISH)

Calculate the remaining days of a coupon’s validity to notify the clients.

SQL script

```
SELECT FLOOR(sysdate - expirationDate) AS  "Remaining days of coupon"
FROM discountP
WHERE FLOOR(sysdate - expirationDate) > 0;
```

Caption of the first five rows of the output



Remaining days of coupon
399
86
257
154

9 Transactions

9.1 *<Salary Raise>*

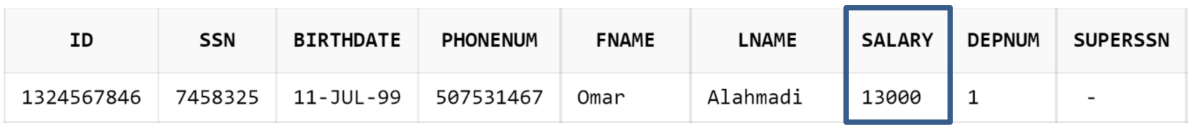
Transaction in natural language (ENGLISH)

Update an entity's information. An update made to an employee's salary will be the demonstration.

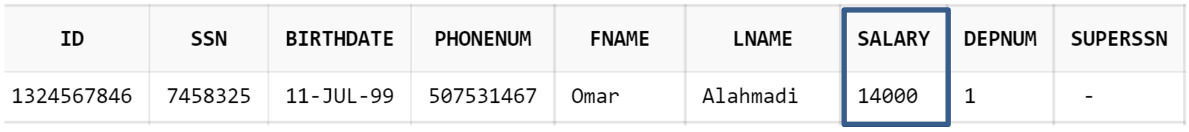
SQL script

UPDATE employee SET salary = 14000 WHERE SSN = 7458325;

Caption of row(s) of the output before and after the update



ID	SSN	BIRTHDATE	PHONENUM	FNAME	LNAME	SALARY	DEPNUM	SUPERSSN
1324567846	7458325	11-JUL-99	507531467	Omar	Alahmadi	13000	1	-



ID	SSN	BIRTHDATE	PHONENUM	FNAME	LNAME	SALARY	DEPNUM	SUPERSSN
1324567846	7458325	11-JUL-99	507531467	Omar	Alahmadi	14000	1	-

APPENDIX

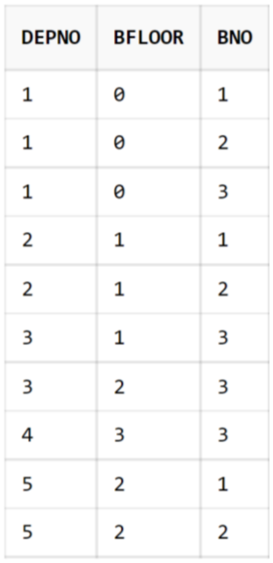
1. Department table

DEPNAME	DEPNUM
Clients Services	1
Design	2
Implementation	3
Decorations	4
Accounting	5

2. Employee table

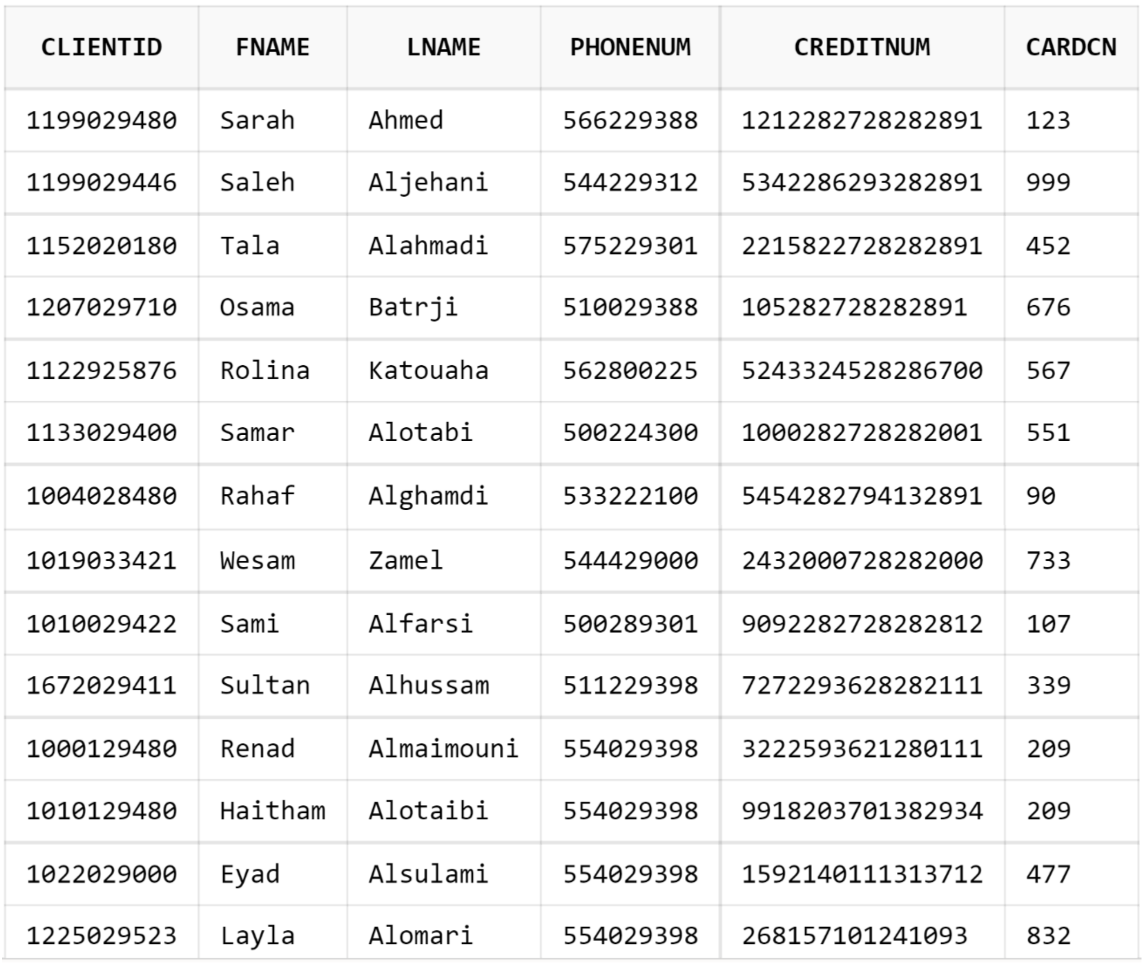
ID	SSN	BIRTHDATE	PHONENUM	FNAME	LNAME	SALARY	DEPNUM	SUPERSSN
1324567846	7458325	11-JUL-99	507531467	Omar	Alahmadi	13000	1	-
1324567847	7479325	07-DEC-99	579980969	Salma	Alotaibi	15000	1	7458325
1126749579	7562577	07-DEC-93	500167903	Talal	Bawazer	12500	1	7479325
1364289769	5735782	06-FEB-93	503863045	Osama	Alomari	1100	1	7458325
1118658920	6734590	01-APR-98	501678424	Sahar	Alkhaldi	16000	2	7458325
1117539379	7625904	03-OCT-92	505103936	Sarah	Alghamdi	13500	2	7479325
1117834902	6783459	05-NOV-99	505103936	Khaled	Alghamdi	14500	2	7479325
1004567845	6789200	27-DEC-91	507101936	Reem	Alamoudi	15000	2	6783459
1010847905	6789673	20-SEP-92	537521570	Maan	Alotaibi	12500	3	6783459
1109873560	7696437	21-AUG-89	500594570	Othman	Algahtani	11500	3	6783459
1112437563	7290137	02-APR-90	500594570	Ahmed	Althunayan	13000	4	6789200
1103437163	7892137	10-DEC-95	511594570	Fahad	Almutairi	9000	4	6789200
1103236163	7807337	01-JAN-96	512094370	Zahra	Alsulami	10000	4	6789200
1103226763	6507337	04-APR-94	531094350	Raghad	Alzahrani	11500	5	5735782
1112136763	7007337	04-APR-00	531094350	Mohammed	Alkhaldi	10500	5	5735782

3. Address table



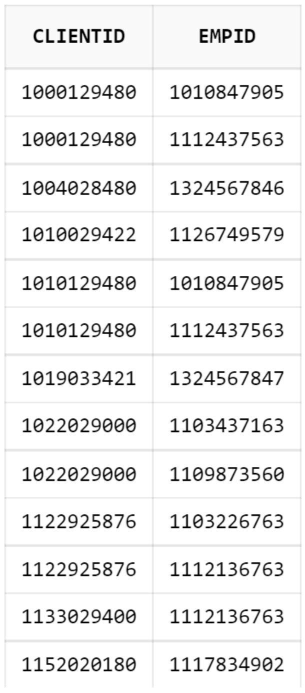
DEPNO	BFLOOR	BNO
1	0	1
1	0	2
1	0	3
2	1	1
2	1	2
3	1	3
3	2	3
4	3	3
5	2	1
5	2	2

4. Client table



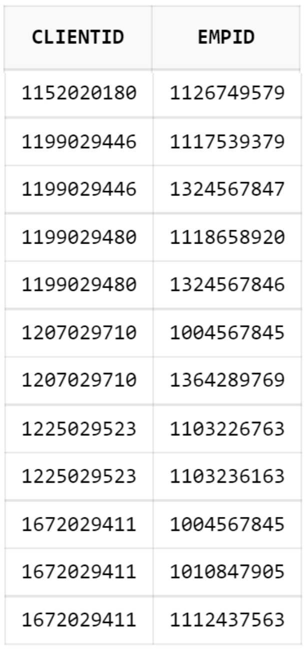
CLIENTID	FNAME	LNAME	PHONENUM	CREDITNUM	CARDCN
1199029480	Sarah	Ahmed	566229388	1212282728282891	123
1199029446	Saleh	Aljehani	544229312	5342286293282891	999
1152020180	Tala	Alahmadi	575229301	2215822728282891	452
1207029710	Osama	Batrji	510029388	105282728282891	676
1122925876	Rolina	Katouaha	562800225	5243324528286700	567
1133029400	Samar	Alotabi	500224300	1000282728282001	551
1004028480	Rahaf	Alghamdi	533222100	5454282794132891	90
1019033421	Wesam	Zamel	544429000	2432000728282000	733
1010029422	Sami	Alfarsi	500289301	9092282728282812	107
1672029411	Sultan	Alhussam	511229398	7272293628282111	339
1000129480	Renad	Almaimouni	554029398	3222593621280111	209
1010129480	Haitham	Alotaibi	554029398	9918203701382934	209
1022029000	Eyad	Alsulami	554029398	1592140111313712	477
1225029523	Layla	Alomari	554029398	268157101241093	832

5. **Assists table**



CLIENTID	EMPID
1000129480	1010847905
1000129480	1112437563
1004028480	1324567846
1010029422	1126749579
1010129480	1010847905
1010129480	1112437563
1019033421	1324567847
1022029000	1103437163
1022029000	1109873560
1122925876	1103226763
1122925876	1112136763
1133029400	1112136763
1152020180	1117834902

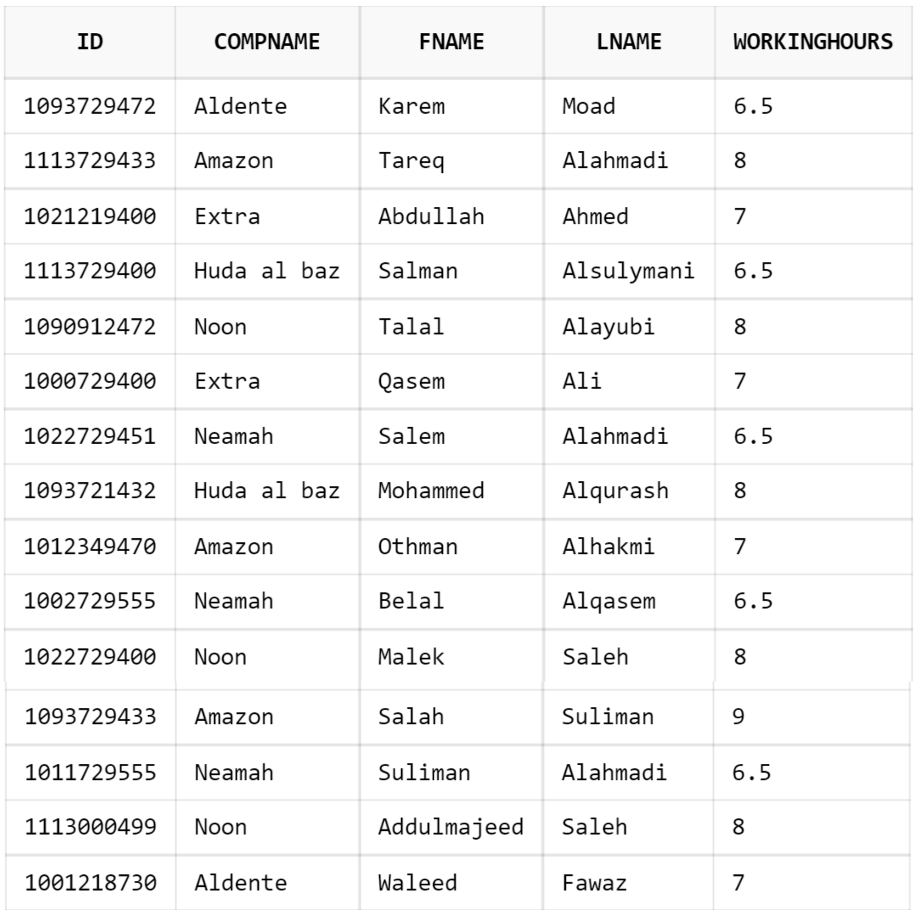
(1)



CLIENTID	EMPID
1152020180	1126749579
1199029446	1117539379
1199029446	1324567847
1199029480	1118658920
1199029480	1324567846
1207029710	1004567845
1207029710	1364289769
1225029523	1103226763
1225029523	1103236163
1672029411	1004567845
1672029411	1010847905
1672029411	1112437563

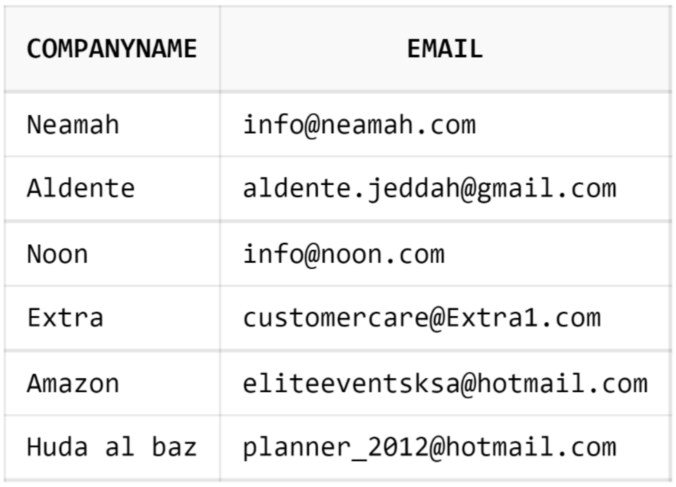
(2)

6. **Driver table**



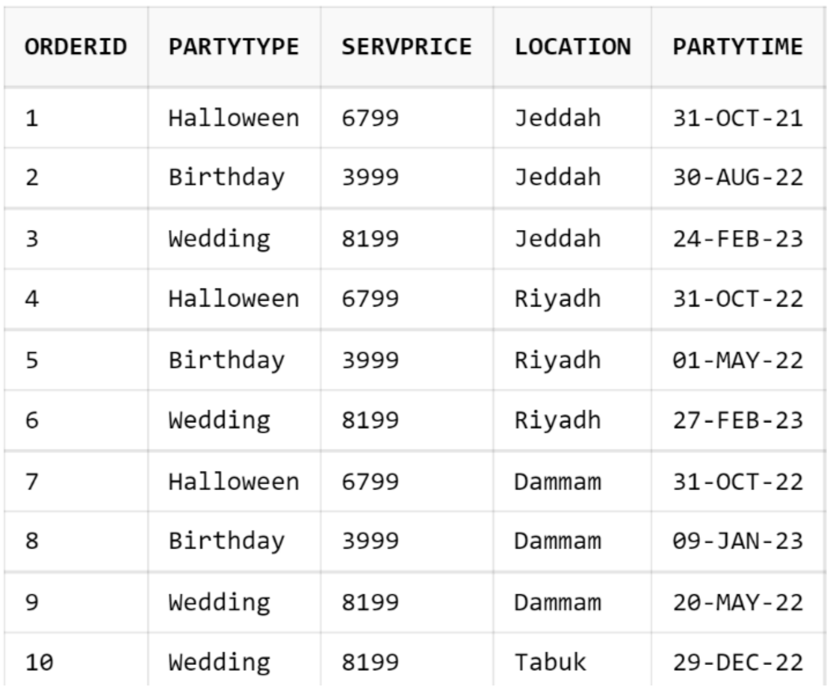
ID	COMPNAME	FNAME	LNAME	WORKINGHOURS
1093729472	Aldente	Karem	Moad	6.5
1113729433	Amazon	Tareq	Alahmadi	8
1021219400	Extra	Abdullah	Ahmed	7
1113729400	Huda al baz	Salman	Alsulymani	6.5
1090912472	Noon	Talal	Alayubi	8
1000729400	Extra	Qasem	Ali	7
1022729451	Neamah	Salem	Alahmadi	6.5
1093721432	Huda al baz	Mohammed	Alqurash	8
1012349470	Amazon	Othman	Alhakmi	7
1002729555	Neamah	Belal	Alqasem	6.5
1022729400	Noon	Malek	Saleh	8
1093729433	Amazon	Salah	Suliman	9
1011729555	Neamah	Suliman	Alahmadi	6.5
1113000499	Noon	Addulmajeed	Saleh	8
1001218730	Aldente	Waleed	Fawaz	7

7. **Supplier table**



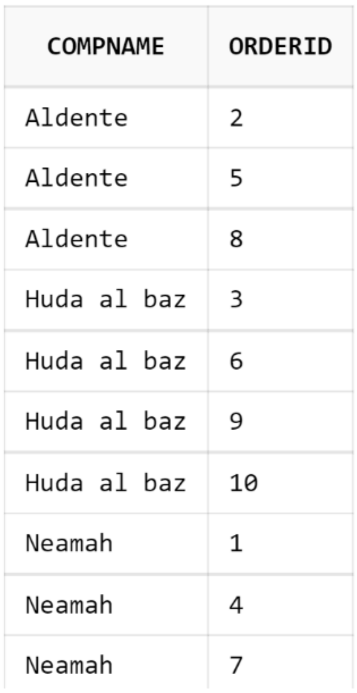
COMPANYNAME	EMAIL
Neamah	info@neamah.com
Aldente	aldente.jeddah@gmail.com
Noon	info@noon.com
Extra	customercare@Extra1.com
Amazon	eliteeventsksa@hotmail.com
Huda al baz	planner_2012@hotmail.com

8. **Party table**



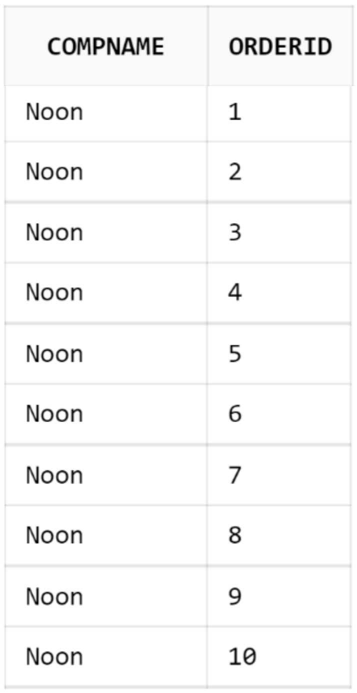
ORDERID	PARTYTYPE	SERVPRICE	LOCATION	PARTYTIME
1	Halloween	6799	Jeddah	31-OCT-21
2	Birthday	3999	Jeddah	30-AUG-22
3	Wedding	8199	Jeddah	24-FEB-23
4	Halloween	6799	Riyadh	31-OCT-22
5	Birthday	3999	Riyadh	01-MAY-22
6	Wedding	8199	Riyadh	27-FEB-23
7	Halloween	6799	Dammam	31-OCT-22
8	Birthday	3999	Dammam	09-JAN-23
9	Wedding	8199	Dammam	20-MAY-22
10	Wedding	8199	Tabuk	29-DEC-22

9. **Provides table**



COMPNAME	ORDERID
Aldente	2
Aldente	5
Aldente	8
Huda al baz	3
Huda al baz	6
Huda al baz	9
Huda al baz	10
Neamah	1
Neamah	4
Neamah	7

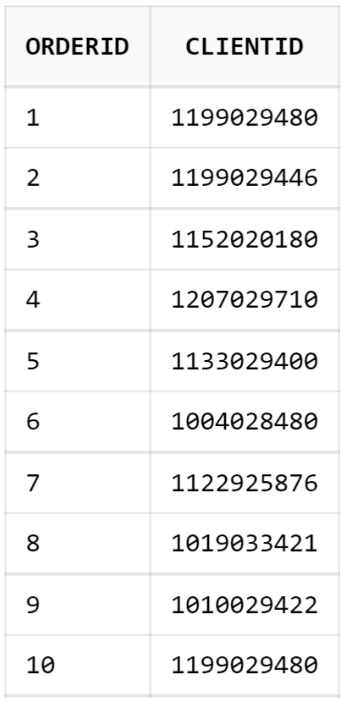
(1)



COMPNAME	ORDERID
Noon	1
Noon	2
Noon	3
Noon	4
Noon	5
Noon	6
Noon	7
Noon	8
Noon	9
Noon	10

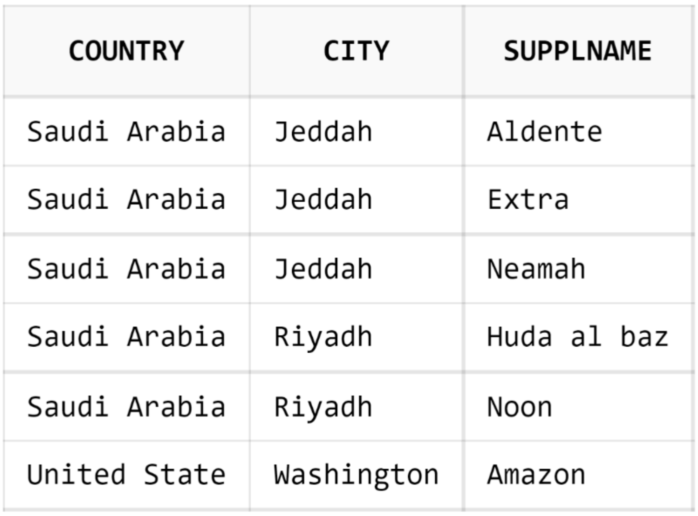
(2)

10. **Chooses table**



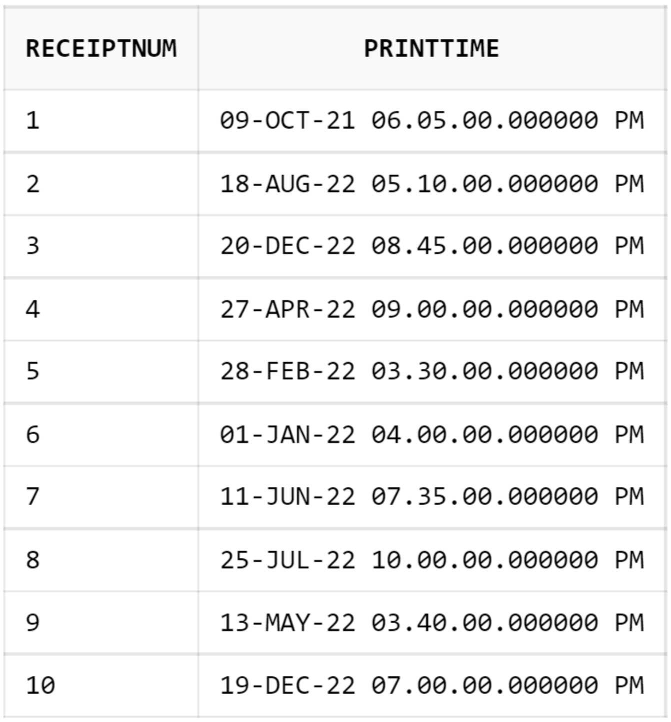
ORDERID	CLIENTID
1	1199029480
2	1199029446
3	1152020180
4	1207029710
5	1133029400
6	1004028480
7	1122925876
8	1019033421
9	1010029422
10	1199029480

11. **Location table**



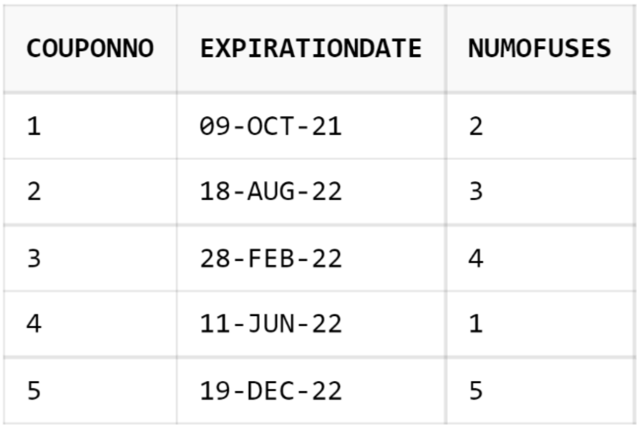
COUNTRY	CITY	SUPPLNAME
Saudi Arabia	Jeddah	Aldente
Saudi Arabia	Jeddah	Extra
Saudi Arabia	Jeddah	Neamah
Saudi Arabia	Riyadh	Huda al baz
Saudi Arabia	Riyadh	Noon
United State	Washington	Amazon

12. **Receipt table**



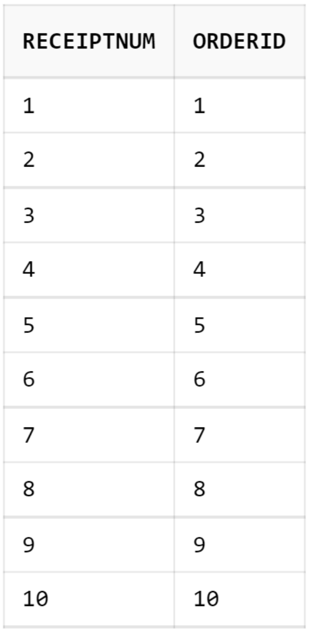
RECEIPTNUM	PRINTTIME
1	09-OCT-21 06.05.00.000000 PM
2	18-AUG-22 05.10.00.000000 PM
3	20-DEC-22 08.45.00.000000 PM
4	27-APR-22 09.00.00.000000 PM
5	28-FEB-22 03.30.00.000000 PM
6	01-JAN-22 04.00.00.000000 PM
7	11-JUN-22 07.35.00.000000 PM
8	25-JUL-22 10.00.00.000000 PM
9	13-MAY-22 03.40.00.000000 PM
10	19-DEC-22 07.00.00.000000 PM

13. **Discount table**



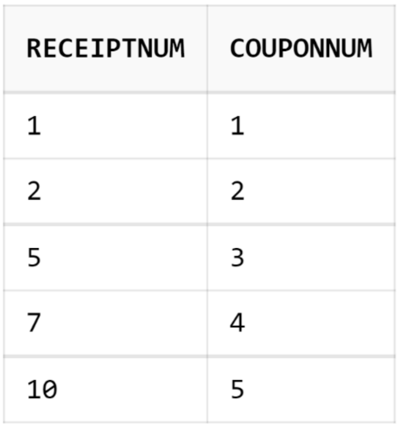
COUPONNO	EXPIRATIONDATE	NUMOFUSES
1	09-OCT-21	2
2	18-AUG-22	3
3	28-FEB-22	4
4	11-JUN-22	1
5	19-DEC-22	5

14. **Affect table**



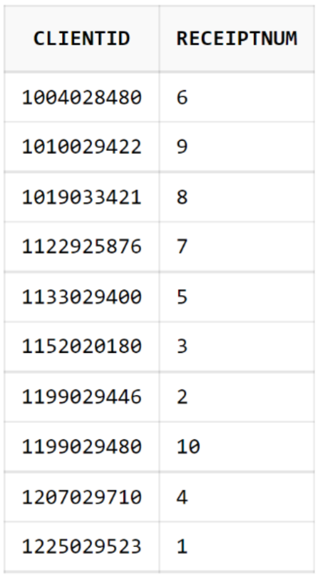
RECEIPTNUM	ORDERID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

15. **Offers table**



RECEIPTNUM	COUPONNUM
1	1
2	2
5	3
7	4
10	5

16. **Has table**



CLIENTID	RECEIPTNUM
1004028480	6
1010029422	9
1019033421	8
1122925876	7
1133029400	5
1152020180	3
1199029446	2
1199029480	10
1207029710	4
1225029523	1