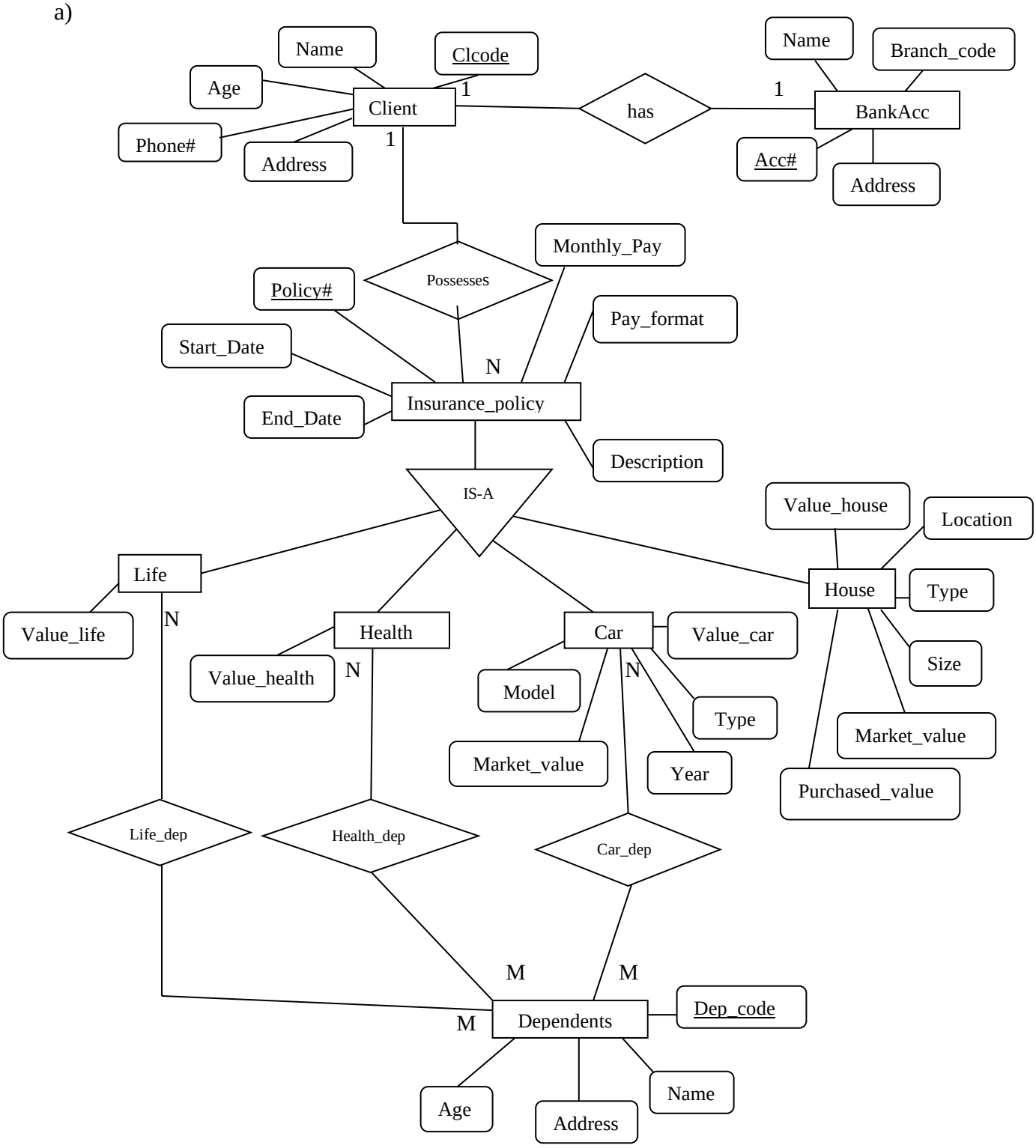
INM370 – Advanced Databases

Tutorial 2 – Object-Relational Databases

Model Answers

a)



b) PRIMARY KEYs *Foreign keys*

Client (Cl_code, Name, Address, Age, Phone#, *Acc#)*
BankAcc (Acc#, Name, Address, Branch_code)
Insurance_Policy (Policy#, Start_Date, End_Date, Description, Monthly_Pay, Pay_format, *Cl_code*)
Dependents (Dep_code, Name, Address, Age)
Life (Policy#, Value_life)
Health (Policy#, Value_health)
Car (Policy#, Value_car, Market_value, Model, Type, Year)
House (Policy#, Value_house, Market_value, Size, Type, Location, Purchased_value)
Life_Dep(Policy#, Dep_Code)
Health_Dep(Policy#, Dep_Code)
Car_Dep(Policy#, Dep_Code)

c)

i. ORDBS Schema using the original SQL:2011 notation

```
/*Type Declarations*/

CREATE TYPE AddressType AS
(
   Street    VARCHAR(20),
   Number    INTEGER,
   City      VARCHAR(15),
   PostCode  VARCHAR(8)
);

CREATE TYPE BankAccType AS
(
   AccNo           INTEGER,
   Branch_Code     VARCHAR(8),
   Name            VARCHAR(20),
   Address         AddressType
);

CREATE TYPE ClientType AS
(
   ClCode    VARCHAR(10),
   Name      VARCHAR(20),
   Address   AddressType,
   Age       INTEGER,
   Phone#    VARCHAR(15),
   Account   REF(BankAccType)
);

CREATE TYPE InsurancePolicyType AS
(
   PolicyNum       VARCHAR(10),
   Start_Date      DATE,
   End_Date        DATE,
   Description     VARCHAR(20),
   Monthly_Pay     DECIMAL(7,2),
   Pay_Format      VARCHAR(30),
   ClientOwner     REF(ClientType)
) NOT FINAL;
```

REF ไปที่ชื่อ Type

```
CREATE TYPE HousePolicyType UNDER InsurancePolicyType AS
(
  Value_House     DECIMAL(9,2),
  Location        AddressType,
  HouseSize       VARCHAR(8),
  HouseType       VARCHAR(8),
  Market_value    DECIMAL(9,2),
  Purchased_Value DECIMAL(9,2),
);

CREATE TYPE CarPolicyType UNDER InsurancePolicyType AS
(
  Value_Car       DECIMAL(7,2),
  Model           VARCHAR(10),
  CarType         VARCHAR(10),
  Market_value    DECIMAL(7,2),
  Year            INTEGER
);

CREATE TYPE HealthPolicyType UNDER InsurancePolicyType AS
(
  Value_Health DECIMAL(9,2)
);

CREATE TYPE LifePolicyType UNDER InsurancePolicyType AS
(
  Value_Life DECIMAL(9,2)
);

CREATE TYPE DependentsType AS
(
  DepCode   VARCHAR(8),
  Name      VARCHAR(20),
  Age       INTEGER,
  Address   AddressType
);

CREATE TYPE LifeDepType AS
(
  LifeInsNum      REF(InsurancePolicyType)
  DepNum          REF(DependentsType)
);

CREATE TYPE HealthDepType AS
(
  HealthInsNum    REF(InsurancePolicyType)
  DepNum          REF(DependentsType));

CREATE TYPE CarDepType AS
(
  CarInsNum       REF(InsurancePolicyType)
  DepNum          REF(DependentsType)
);
```

```
/*Table Declarations*/

CREATE TABLE BankAcc OF BankAccType
(
   PRIMARY KEY(AccNo)
);


CREATE TABLE Client OF ClientType
(
   PRIMARY KEY (ClCode),
   Account SCOPE BankAcc
);

CREATE TABLE InsurancePolicy OF InsurancePolicyType
(
   PRIMARY KEY (PolicyNum),
   ClientOwner SCOPE Client REFERENCES ARE CHECKED
   // One could add  REFERENCES ARE CHECKED to indicate that referential integrity is ensured. See p321 in
   //the main textbook. One could add it in other places too. The alternative is to state REFERENCES ARE NOT
   //CHECKED.
);

CREATE TABLE House OF HousePolicyType UNDER InsurancePolicy;

CREATE TABLE Car OF CarPolicyType UNDER InsurancePolicy;

CREATE TABLE Health OF HealthPolicyType UNDER InsurancePolicy;

CREATE TABLE Life OF LifePolicyType UNDER InsurancePolicy;

CREATE TABLE Dependents OF DependentsType
(
   PRIMARY KEY (DepCode)
);
```

SCOPE ตามด้วย TableName

Var ตัวนึงของ OF TYPE

Life_Dep(Policy#, Dep_Code)

```
CREATE TABLE LifeDep OF LifeDepType
(
   PRIMARY KEY (LifeInsNum, DepNum)
   LifeInsNum      SCOPE Life,
   DepNum          SCOPE Dependents
);

CREATE TABLE HealthDep OF HealthDepType
(
   PRIMARY KEY (HealthInsNum, DepNum)
   HealthInsNum    SCOPE Health,
   DepNum          SCOPE Dependents
);

CREATE TABLE CarDep OF CarDepType
(
   PRIMARY KEY (CarInsNum, DepNum)
   CarInsNum       SCOPE Car,
   DepNum          SCOPE Dependents
);
```

ii. ORDBS Schema using SQL:2011 notation implemented by Oracle

```
/*Type Declarations*/

CREATE TYPE AddressType AS OBJECT
(
  Street    VARCHAR2(20),
  Num       NUMBER,
  City      VARCHAR2(15),
  PostCode  VARCHAR2(8)
);

CREATE TYPE BankAccType AS OBJECT
(
  AccNo           VARCHAR2(5),
  Branch_Code     VARCHAR2(8),
  Name            VARCHAR2(20),
  Address         AddressType
);

CREATE TYPE ClientType AS OBJECT
(
  ClCode    VARCHAR2(10),
  Name      VARCHAR2(20),
  Address   AddressType,
  Age       NUMBER,
  Phone#    VARCHAR2(15),
  Account REF BankAccType
);

CREATE TYPE InsurancePolicyType AS OBJECT
(
  PolicyNum       VARCHAR2(10),
  Start_Date      DATE,
  End_Date        DATE,
  Description     VARCHAR2(20),
  Monthly_Pay     NUMBER (7,2),
  Pay_Format      VARCHAR2(30),
  ClientOwner REF ClientType
) NOT FINAL;

CREATE TYPE HousePolicyType UNDER InsurancePolicyType
(
  Value_House     NUMBER (9,2),
  Location        AddressType,
  HouseSize       VARCHAR2(8),
  HouseType       VARCHAR2(8),
  Market_Value    NUMBER (9,2),
  Purchased_Value NUMBER(9,2)
);

CREATE TYPE CarPolicyType UNDER InsurancePolicyType
(
  Value_Car       NUMBER(7,2),
  Model           VARCHAR2(10),
  CarType         VARCHAR2(10),
  Market_value    NUMBER(7,2),
```

```
  Year            NUMBER
);

CREATE TYPE HealthPolicyType UNDER InsurancePolicyType
(
  Value_Health NUMBER
);

CREATE TYPE LifePolicyType UNDER InsurancePolicyType
(
  Value_Life NUMBER
);

CREATE TYPE DependentsType AS OBJECT
(
  DepCode   VARCHAR2(8),
  Name      VARCHAR2(20),
  Age       NUMBER,
  Address   AddressType
);

CREATE TYPE LifeDepType AS OBJECT
(
  LifeInsNum      REF InsurancePolicyType,
  DepNum          REF DependentsType
);

CREATE TYPE HealthDepType AS OBJECT
(
  HealthInsNum    REF InsurancePolicyType,
  DepNum          REF DependentsType
);

CREATE TYPE CarDepType AS OBJECT
(
  CarInsNum       REF InsurancePolicyType,
  DepNum          REF DependentsType
);
```

/*Table Declarations*/

```
CREATE TABLE BankAcc OF BankAccType
(
  AccNo PRIMARY KEY
);

CREATE TABLE Client of ClientType
(
  ClCode    PRIMARY KEY,
  SCOPE FOR (Account) IS BankAcc
);

CREATE TABLE InsurancePolicy OF InsurancePolicyType
(
  PolicyNum PRIMARY KEY,
  SCOPE FOR (ClientOwner) IS Client
);
```

```
CREATE TABLE House OF HousePolicyType;

CREATE TABLE Car OF CarPolicyType;

CREATE TABLE Health OF HealthPolicyType;

CREATE TABLE Life OF LifePolicyType;

CREATE TABLE Dependents OF DependentsType
(
  DepCode PRIMARY KEY
);

CREATE TABLE LifeDep OF LifeDepType
(
  SCOPE FOR (LifeInsNum) IS Life,
  SCOPE FOR (DepNum) IS Dependents
);

CREATE TABLE HealthDep of HealthDepType
(
  SCOPE FOR (HealthInsNum) IS Health,
  SCOPE FOR (DepNum) IS Dependents
);
CREATE TABLE CarDep of CarDepType
(
  SCOPE FOR (CarInsNum) is Car,
  SCOPE FOR (DepNum) is Dependents
);
```

d)

i) original SQL:2011 notation

1)
```
SELECT      Client.*, Client.Account->Name AS BankName
FROM        Client;
```

2)
```
SELECT      Client.Name, InsurancePolicy.PolicyNum
FROM        Client, InsurancePolicy
WHERE       Client.ClCode = InsurancePolicy.ClientOwner->ClCode;
```
But, this approach (where an expensive JOIN is created between the two tables) defeats the purpose of using ORDB approach (which avoids expensive JOINs).

Instead, one should use object-relational approach:
```
SELECT      InsurancePolicy.ClientOwner->Name, InsurancePolicy.PolicyNum
FROM        InsurancePolicy
```

3)
```
SELECT      Life.PolicyNum, Dependents.Name
FROM        Life, LifeDep, Dependents
WHERE       Life.PolicyNum = LifeDep.LifeInsNum->PolicyNum
AND         Dependents.DepCode = LifeDep.DepNum->DepCode;
```

Instead, one can use object-relational approach:

```
SELECT      ld.LifeInsNum->PolicyNum, ld.DepNum->Name
FROM        LifeDep ld
```

4)

```
SELECT      Client.Name
FROM        Client
WHERE       Client.Account->Address.City = 'London';
```

ii) **SQL:2011 notation implemented by Oracle**

1)

```
SELECT      c.*, c.Account.Name AS BankName
FROM        Client c;
```

or

```
SELECT      c.*, DEREF(c.Account).Name AS BankName
FROM        Client c;
```

2)

```
SELECT      c.Name, ip.PolicyNum
FROM        Client c, InsurancePolicy ip
WHERE       c.ClCode = ip.ClientOwner.ClCode;
```

or

```
SELECT      c.Name, ip.PolicyNum
FROM        Client c, InsurancePolicy ip
WHERE       c.ClCode = DEREF(ip.ClientOwner).ClCode;
```

Instead, one can use object-relational approach:

```
SELECT      ip.ClientOwner.Name, ip.PolicyNum
FROM        InsurancePolicy ip;
```

or

```
SELECT      DEREF(ip.ClientOwner).Name, ip.PolicyNum
FROM        InsurancePolicy ip
```

3)

```
SELECT      l.PolicyNum, d.Name
FROM        Life l, LifeDep ld, Dependents d
WHERE       l.PolicyNum = ld.LifeInsNum.PolicyNum
AND         d.DepCode = ld.DepNum.DepCode;
```

or

```
SELECT      l.PolicyNum, d.Name
FROM        Life l, LifeDep ld, Dependents d
WHERE       l.PolicyNum = DEREF(ld.LifeInsNum).PolicyNum
AND         d.DepCode = DEREF(ld.DepNum).DepCode;
```

Instead, one can use object-relational approach:

```
SELECT      ld.LifeInsNum.PolicyNum, ld.DepNum.Name
FROM        LifeDep ld
```

or

```
SELECT      DEREF(ld.LifeInsNum).PolicyNum, DEREF(ld.DepNum).Name
FROM        LifeDep ld
```

4)

```
SELECT      c.Name
FROM        Client c
WHERE       c.Account.Address.City = 'London';
```

or

```
SELECT      c.Name
FROM        Client c
WHERE       DEREF(c.Account).Address.City = 'London';
```