UNIVERSITY OF CAMBRIDGE

Welcome

Course Title:	**Relational Database Design**
Email:	**service-desk@ucs.cam.ac.uk**

UCS

Topics covered:

- Entity-Relationship Diagrams
  - Tables and Fields
  - Keys and Joins
  - A practical understanding of normalisation

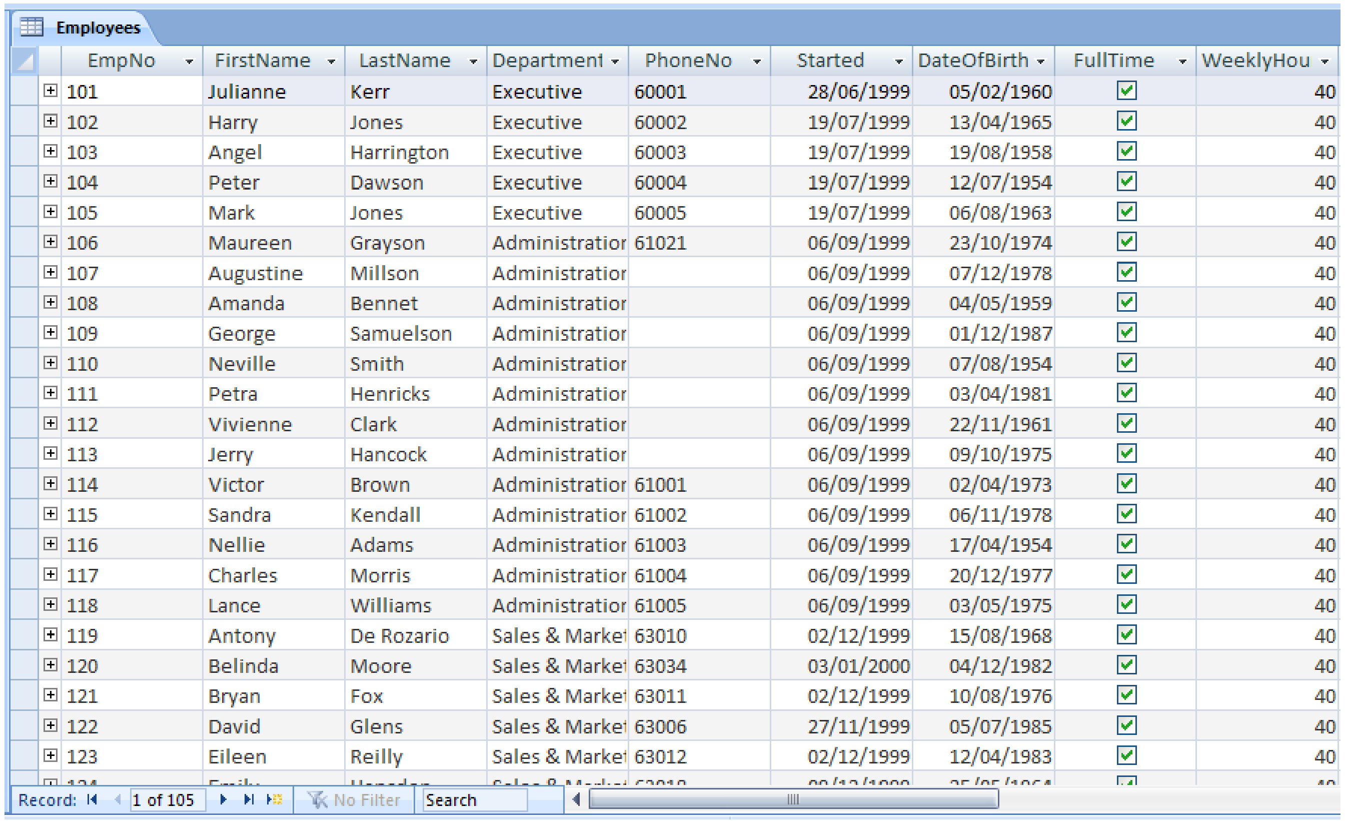
Lesson Objectives

By the end of this session you will be able to:

- ✓ Define: database, table, field
- ✓ Explain join types – 1:1, 1:n, m:n
- ✓ Apply Primary and Foreign keys to tables
- ✓ Draw an ERD for a given scenario

Ask questions at any time please!

What is a database?



Employees

	EmpNo	FirstName	LastName	Department	PhoneNo	Started	DateOfBirth	FullTime	WeeklyHou
⊞	101	Julianne	Kerr	Executive	60001	28/06/1999	05/02/1960	✓	40
⊞	102	Harry	Jones	Executive	60002	19/07/1999	13/04/1965	✓	40
⊞	103	Angel	Harrington	Executive	60003	19/07/1999	19/08/1958	✓	40
⊞	104	Peter	Dawson	Executive	60004	19/07/1999	12/07/1954	✓	40
⊞	105	Mark	Jones	Executive	60005	19/07/1999	06/08/1963	✓	40
⊞	106	Maureen	Grayson	Administratior	61021	06/09/1999	23/10/1974	✓	40
⊞	107	Augustine	Millson	Administratior		06/09/1999	07/12/1978	✓	40
⊞	108	Amanda	Bennet	Administratior		06/09/1999	04/05/1959	✓	40
⊞	109	George	Samuelson	Administratior		06/09/1999	01/12/1987	✓	40
⊞	110	Neville	Smith	Administratior		06/09/1999	07/08/1954	✓	40
⊞	111	Petra	Henricks	Administratior		06/09/1999	03/04/1981	✓	40
⊞	112	Vivienne	Clark	Administratior		06/09/1999	22/11/1961	✓	40
⊞	113	Jerry	Hancock	Administratior		06/09/1999	09/10/1975	✓	40
⊞	114	Victor	Brown	Administratior	61001	06/09/1999	02/04/1973	✓	40
⊞	115	Sandra	Kendall	Administratior	61002	06/09/1999	06/11/1978	✓	40
⊞	116	Nellie	Adams	Administratior	61003	06/09/1999	17/04/1954	✓	40
⊞	117	Charles	Morris	Administratior	61004	06/09/1999	20/12/1977	✓	40
⊞	118	Lance	Williams	Administratior	61005	06/09/1999	03/05/1975	✓	40
⊞	119	Antony	De Rozario	Sales & Market	63010	02/12/1999	15/08/1968	✓	40
⊞	120	Belinda	Moore	Sales & Market	63034	03/01/2000	04/12/1982	✓	40
⊞	121	Bryan	Fox	Sales & Market	63011	02/12/1999	10/08/1976	✓	40
⊞	122	David	Glens	Sales & Market	63006	27/11/1999	05/07/1985	✓	40
⊞	123	Eileen	Reilly	Sales & Market	63012	02/12/1999	12/04/1983	✓	40
⊞	124	Emily	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	✓	[illegible]

Record: 1 of 105 No Filter Search

A database is a **centralised** and **structured set of data** stored on a computer system

It provides facilities for **adding**, **modifying** and **deleting** the data when required

It also provides facilities for transforming **queried data** into useful **information**

RDB Development Overview

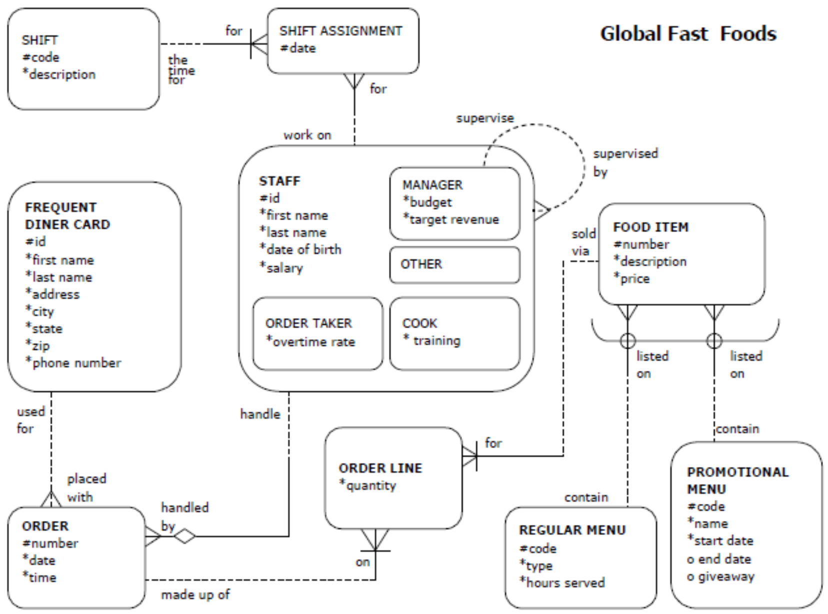
1	Understand the requirements: What data do you want to store? What information do you want to retrieve?
2	Produce an entity relationship diagram
3	Implement in software, e.g: Access (Up to 2 GB max size) MySQL (2TB max) PostGres (unlimited size) Oracle (unlimited size) etc.

Entity Relationship Design Process

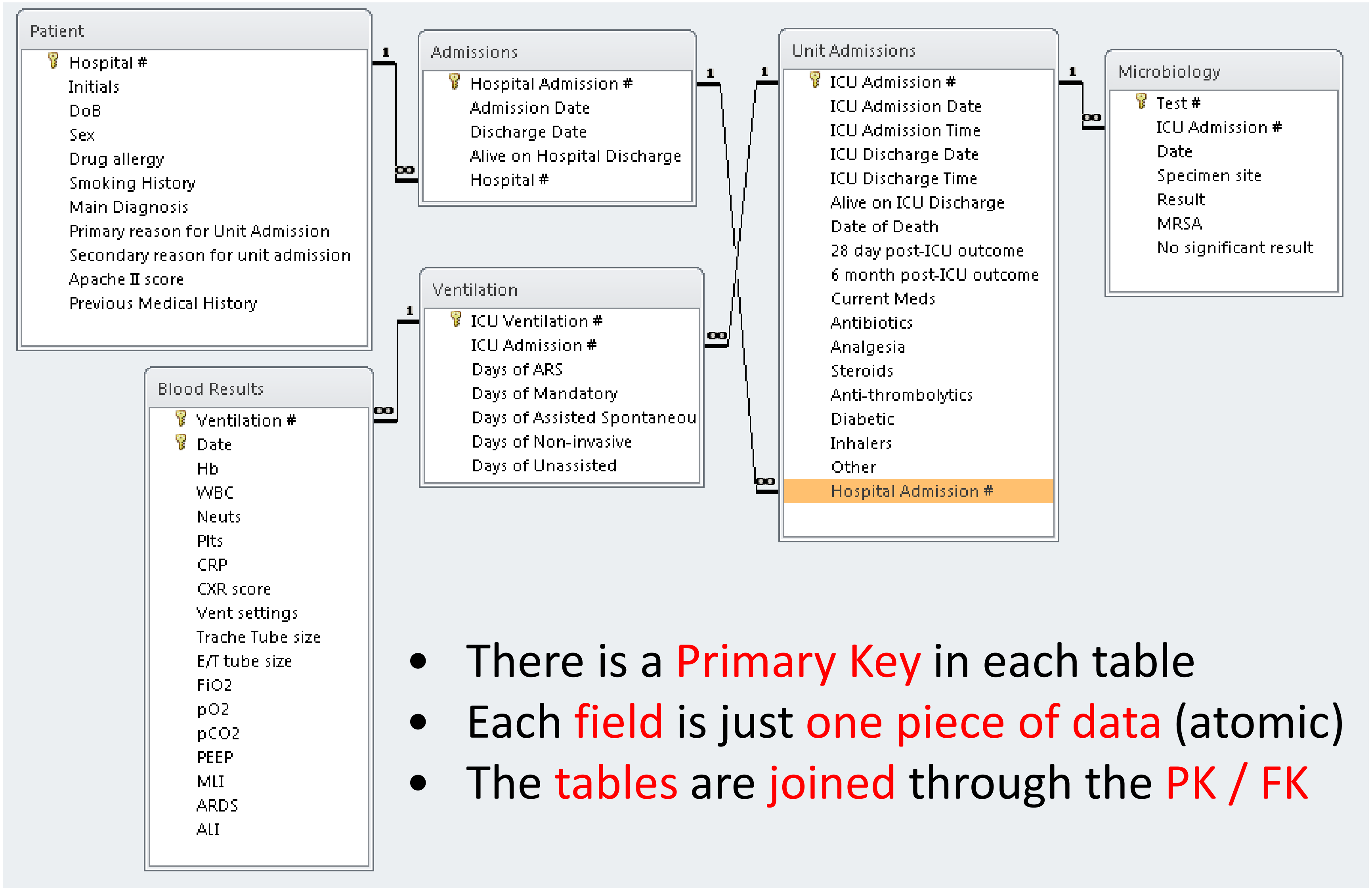
Step 1 – Identify the data

Step 2 – Group the data into tables and allocate a primary key for each table

Step 3 – Join the tables using PK to FK



Tables (entities) hold groups of fields (attributes)



Patient
- Hospital #
- Initials
- DoB
- Sex
- Drug allergy
- Smoking History
- Main Diagnosis
- Primary reason for Unit Admission
- Secondary reason for unit admission
- Apache II score
- Previous Medical History

Admissions
- Hospital Admission #
- Admission Date
- Discharge Date
- Alive on Hospital Discharge
- Hospital #

Unit Admissions
- ICU Admission #
- ICU Admission Date
- ICU Admission Time
- ICU Discharge Date
- ICU Discharge Time
- Alive on ICU Discharge
- Date of Death
- 28 day post-ICU outcome
- 6 month post-ICU outcome
- Current Meds
- Antibiotics
- Analgesia
- Steroids
- Anti-thrombolytics
- Diabetic
- Inhalers
- Other
- Hospital Admission #

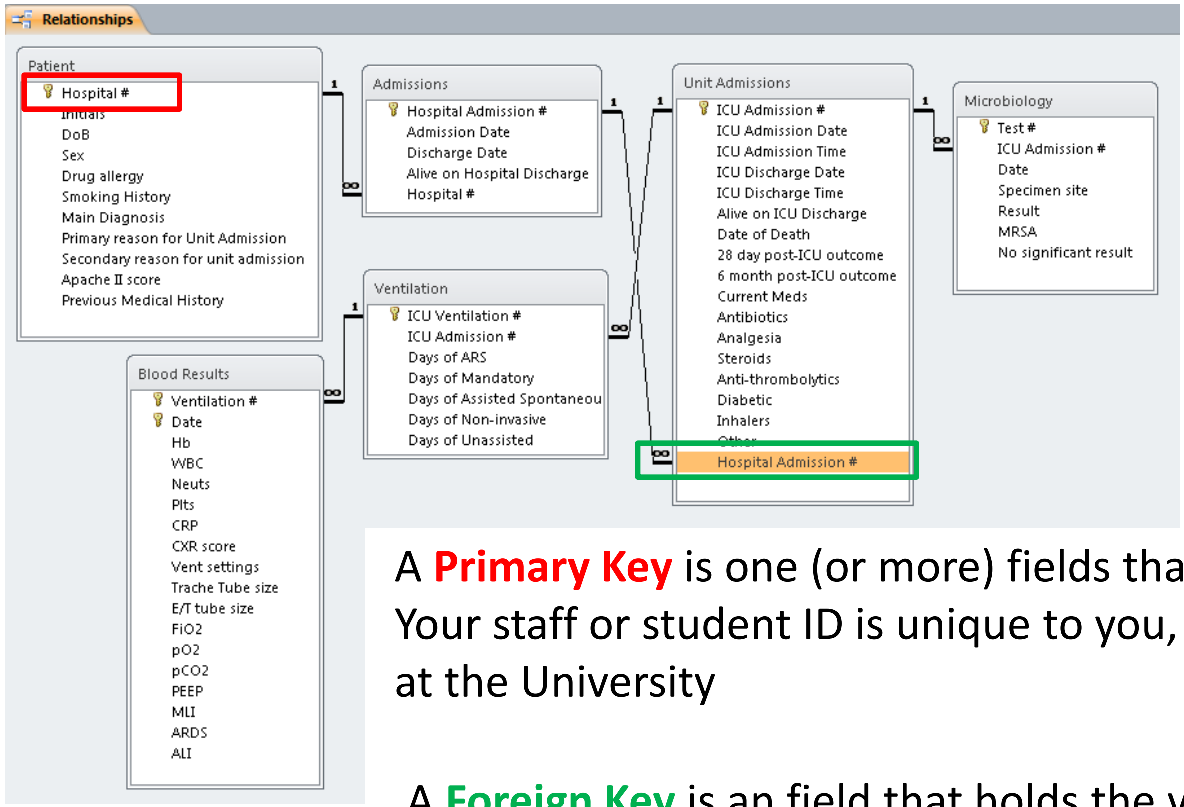
Microbiology
- Test #
- ICU Admission #
- Date
- Specimen site
- Result
- MRSA
- No significant result

Ventilation
- ICU Ventilation #
- ICU Admission #
- Days of ARS
- Days of Mandatory
- Days of Assisted Spontaneou
- Days of Non-invasive
- Days of Unassisted

Blood Results
- Ventilation #
- Date
- Hb
- WBC
- Neuts
- Plts
- CRP
- CXR score
- Vent settings
- Trache Tube size
- E/T tube size
- FiO2
- pO2
- pCO2
- PEEP
- MLI
- ARDS
- ALI

- There is a Primary Key in each table
- Each field is just one piece of data (atomic)
- The tables are joined through the PK / FK

What are Primary and Foreign Keys?

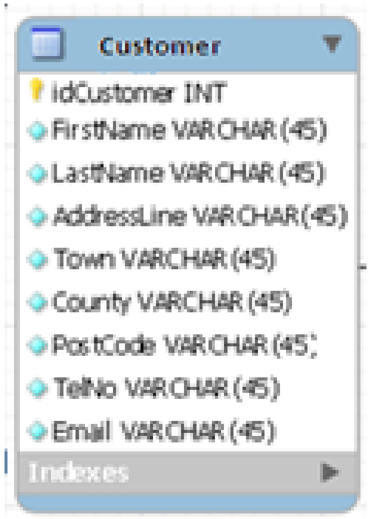


A **Primary Key** is one (or more) fields that uniquely identify a record
Your staff or student ID is unique to you, this is the PK field in your record at the University

A **Foreign Key** is an field that holds the value of a primary key of another table

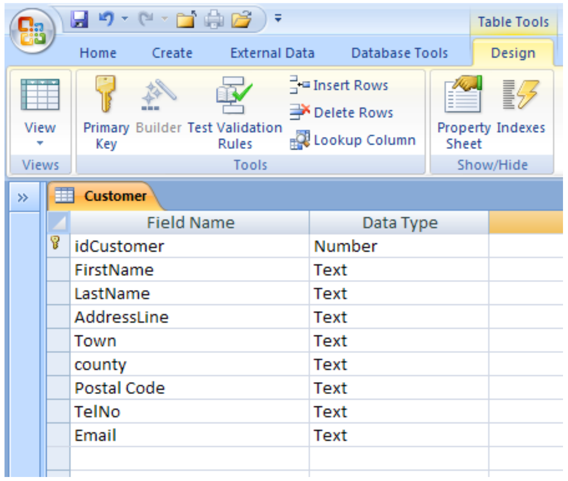
Common fields are used to join tables
joins are either PK to FK (1:n) or PK to PK (1:1)

How Does the ERD relate to the Implementation?



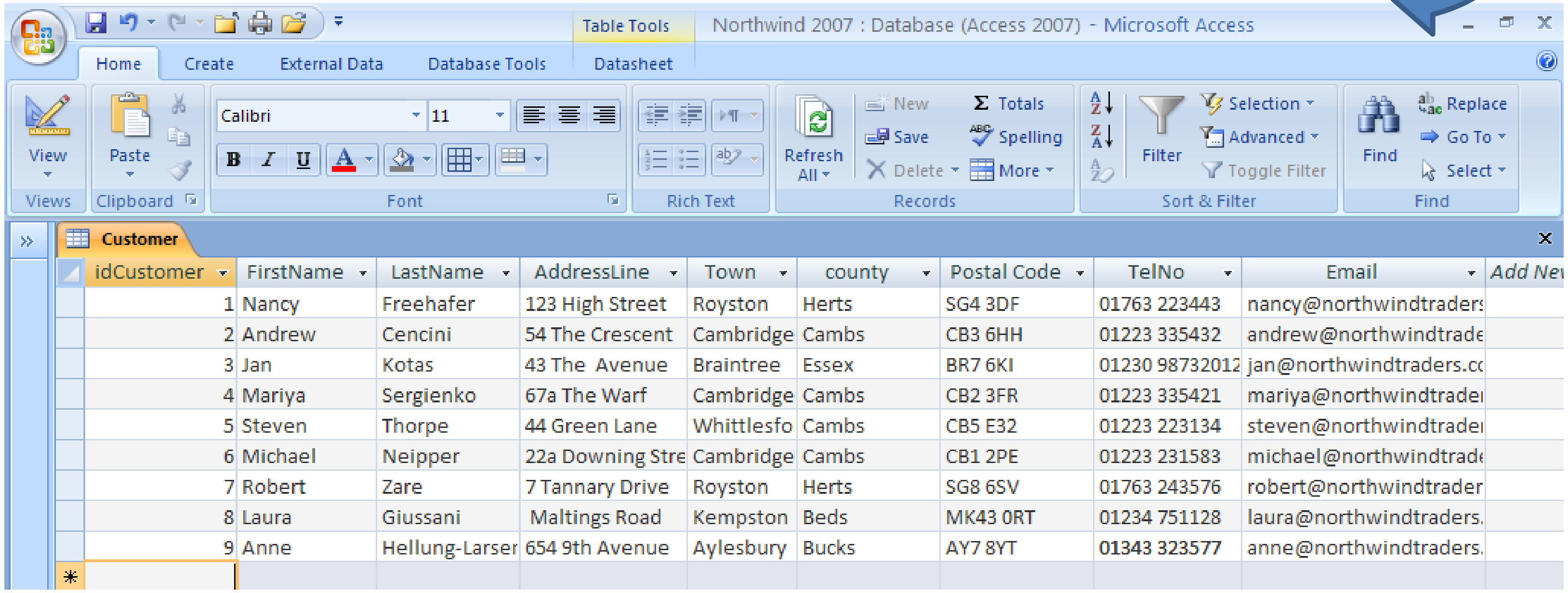
Customer

- idCustomer INT
- FirstName VARCHAR(45)
- LastName VARCHAR(45)
- AddressLine VARCHAR(45)
- Town VARCHAR(45)
- County VARCHAR(45)
- PostCode VARCHAR(45)
- TelNo VARCHAR(45)
- Email VARCHAR(45)



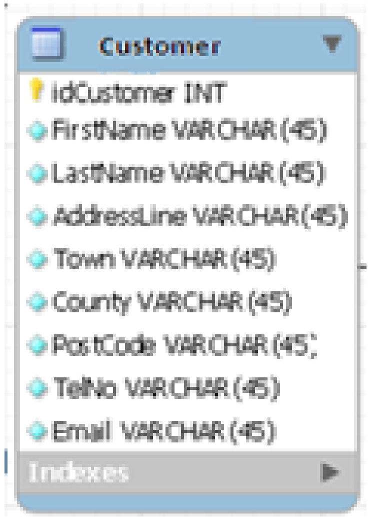
Field Name	Data Type
idCustomer	Number
FirstName	Text
LastName	Text
AddressLine	Text
Town	Text
county	Text
Postal Code	Text
TelNo	Text
Email	Text

MS_Access



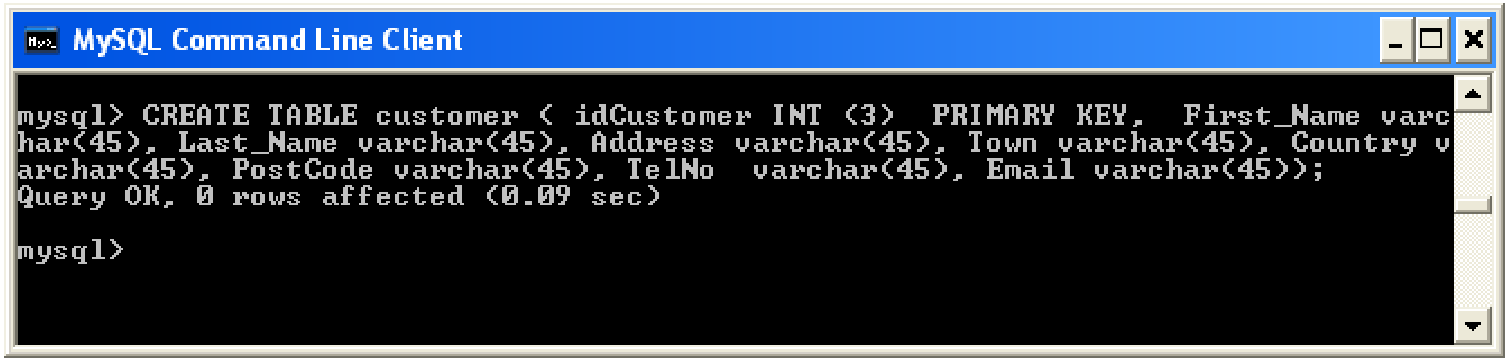
idCustomer	FirstName	LastName	AddressLine	Town	county	Postal Code	TelNo	Email	Add Ne
1	Nancy	Freehafer	123 High Street	Royston	Herts	SG4 3DF	01763 223443	nancy@northwindtraders	
2	Andrew	Cencini	54 The Crescent	Cambridge	Cambs	CB3 6HH	01223 335432	andrew@northwindtrade	
3	Jan	Kotas	43 The Avenue	Braintree	Essex	BR7 6KI	01230 98732012	jan@northwindtraders.cc	
4	Mariya	Sergienko	67a The Warf	Cambridge	Cambs	CB2 3FR	01223 335421	mariya@northwindtrader	
5	Steven	Thorpe	44 Green Lane	Whittlesfo	Cambs	CB5 E32	01223 223134	steven@northwindtrader	
6	Michael	Neipper	22a Downing Stre	Cambridge	Cambs	CB1 2PE	01223 231583	michael@northwindtrade	
7	Robert	Zare	7 Tannary Drive	Royston	Herts	SG8 6SV	01763 243576	robert@northwindtrader	
8	Laura	Giussani	Maltings Road	Kempston	Beds	MK43 0RT	01234 751128	laura@northwindtraders.	
9	Anne	Hellung-Larser	654 9th Avenue	Aylesbury	Bucks	AY7 8YT	01343 323577	anne@northwindtraders.	

. . . or in MySQL



$\Rightarrow$

```
CREATE TABLE customer
(
idCustomer INT (3)
PRIMARY KEY,
First_Name varchar(45),
Last_Name varchar(45),
Address varchar(45),
Town varchar(45),
Country varchar(45),
PostCode varchar(45),
TelNo  varchar(45),
Email varchar(45)
);
```

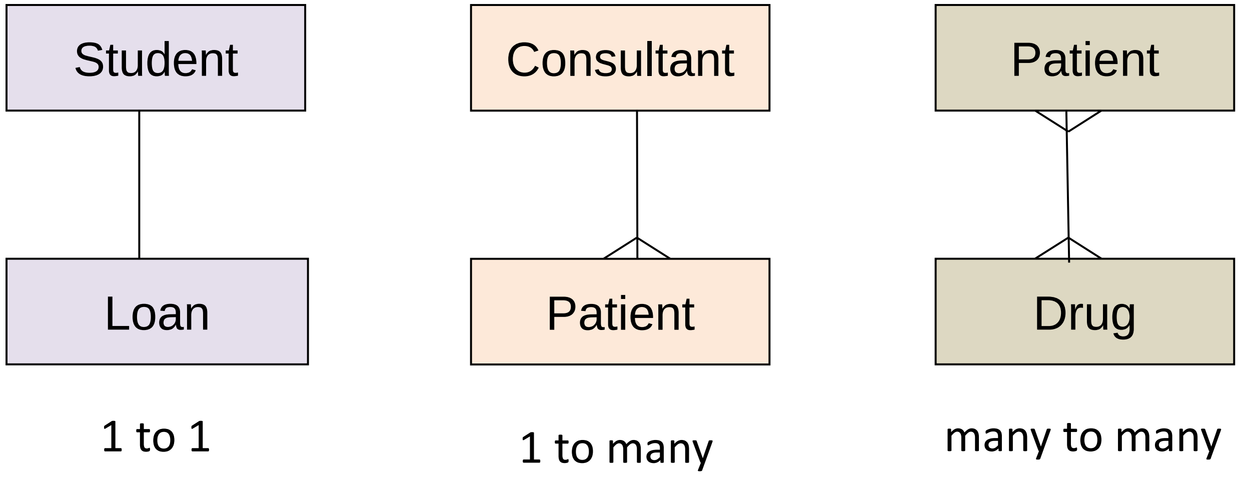


```
mysql> CREATE TABLE customer ( idCustomer INT (3)  PRIMARY KEY,  First_Name varc
har(45), Last_Name varchar(45), Address varchar(45), Town varchar(45), Country v
archar(45), PostCode varchar(45), TelNo  varchar(45), Email varchar(45));
Query OK, 0 rows affected (0.09 sec)

mysql>
```

Going back to Join Types

Relationships are drawn with a single end (**1**) or a crow's foot (**many**):

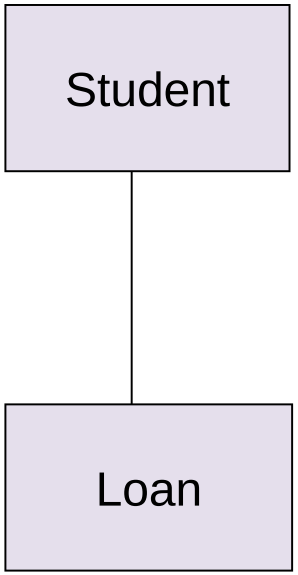


Each Student has one Loan
Each Loan is allocated to one Student

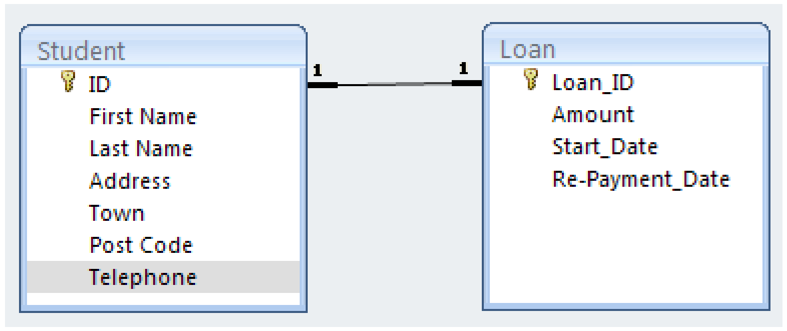
Each Consultant has many Patients
Each Patient is allocated to one Consultant

Each Patient may be prescribed many Drugs
Each Drug may be prescribed to many Patients

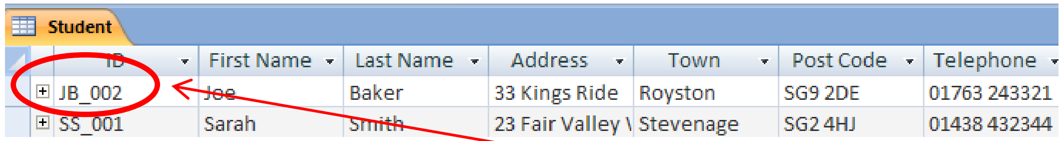
One to one (1:1)



The **name** of the PK field does **not need to match**



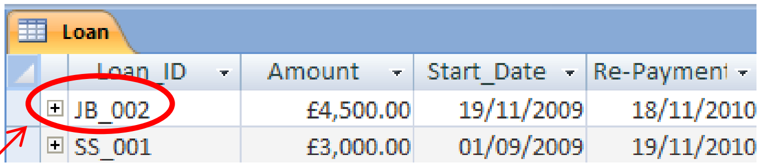
Each Student has one Loan



Student

ID	First Name	Last Name	Address	Town	Post Code	Telephone
JB_002	Joe	Baker	33 Kings Ride	Royston	SG9 2DE	01763 243321
SS_001	Sarah	Smith	23 Fair Valley \	Stevenage	SG2 4HJ	01438 432344

Each Loan is allocated to one Student



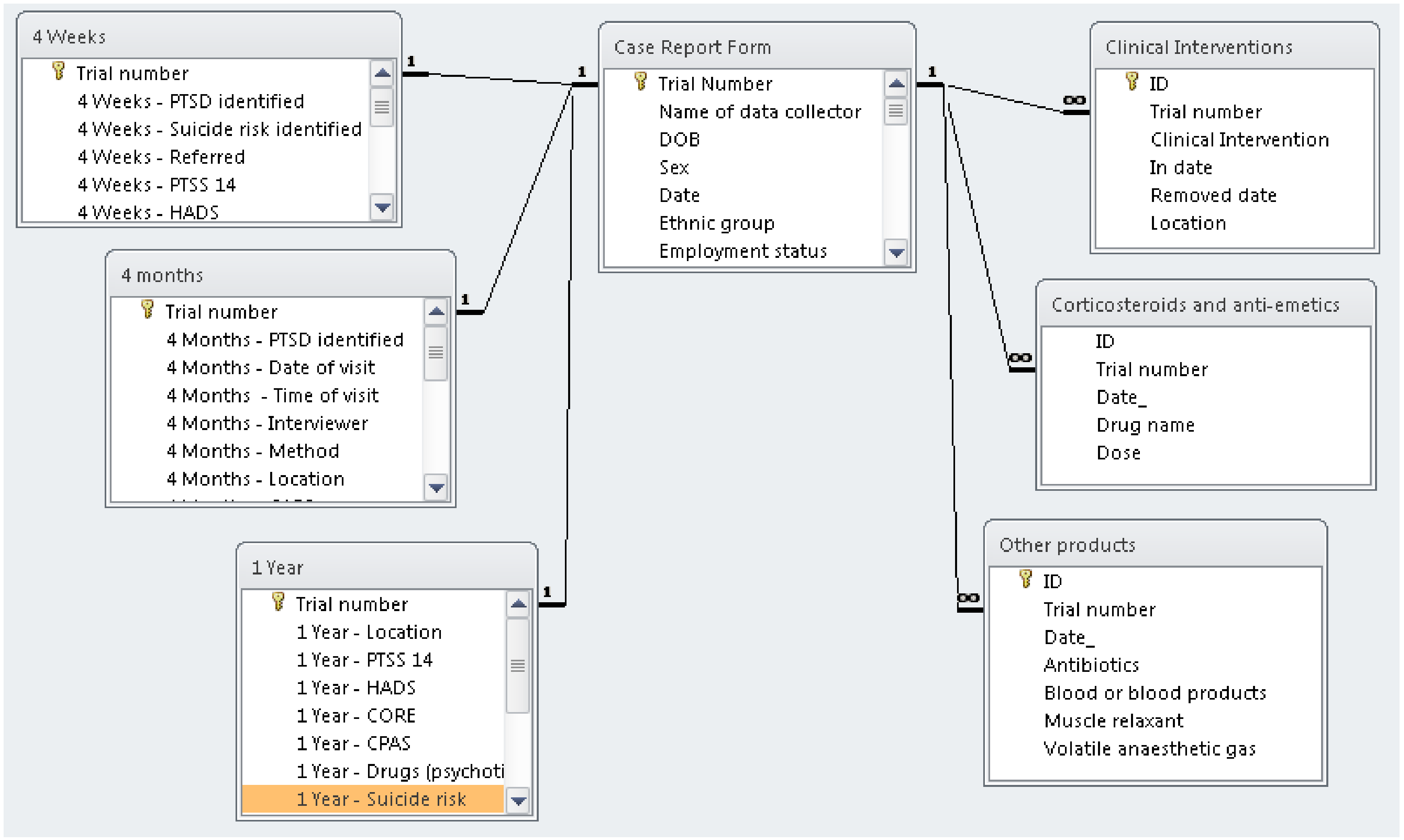
Loan

Loan_ID	Amount	Start_Date	Re-Payment
JB_002	£4,500.00	19/11/2009	18/11/2010
SS_001	£3,000.00	01/09/2009	19/11/2010

Each PK is unique in each table and the PK values must match in both tables.

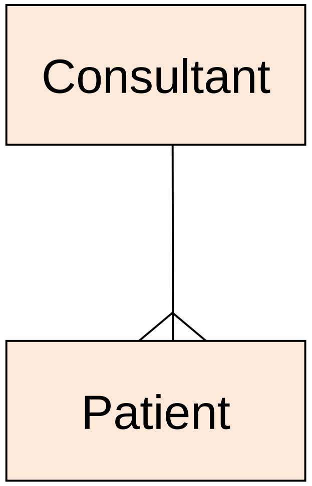
Example usage for 1:1

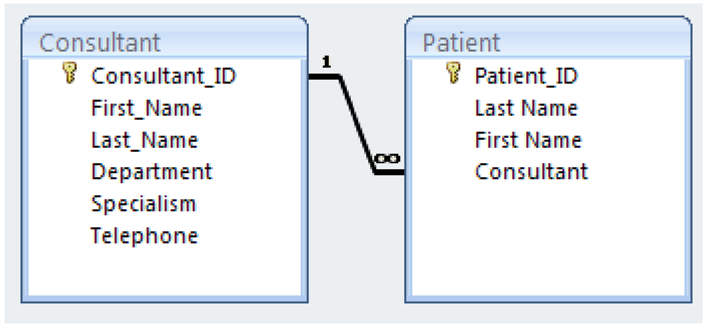
Microsoft Access Database



A 'Case Report' patient takes each survey once over a period of time

One to Many (1:n)

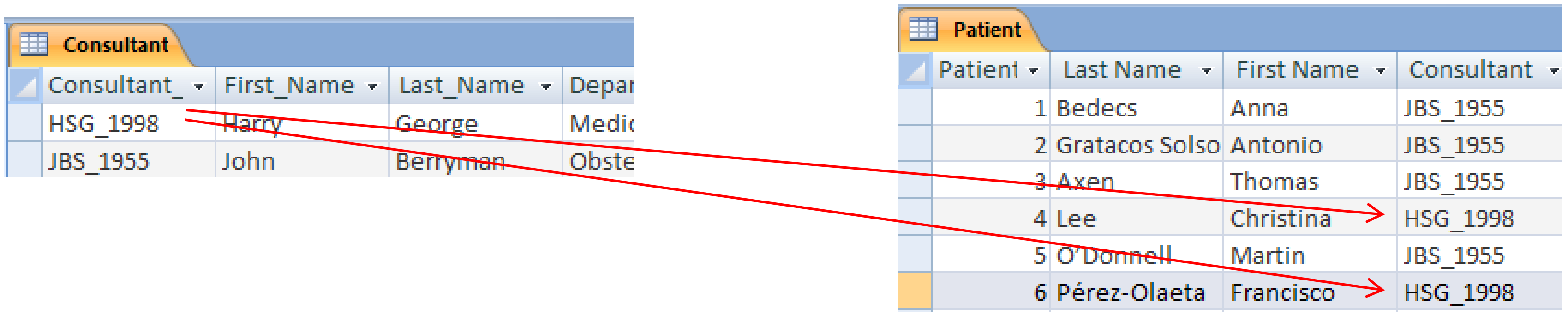




Consultant	Patient
Consultant_ID	Patient_ID
First_Name	Last Name
Last_Name	First Name
Department	Consultant
Specialism	
Telephone	

Each Consultant has many Patients

Each Patient is allocated to one Consultant



Consultant

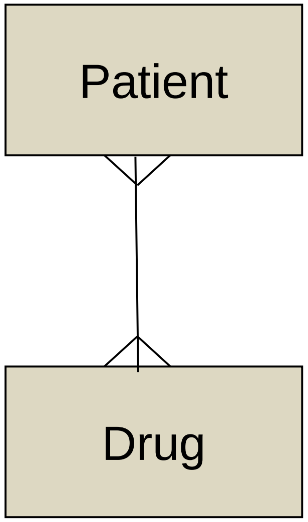
Consultant_	First_Name	Last_Name	Depar
HSG_1998	Harry	George	Medi
JBS_1955	John	Berryman	Obste

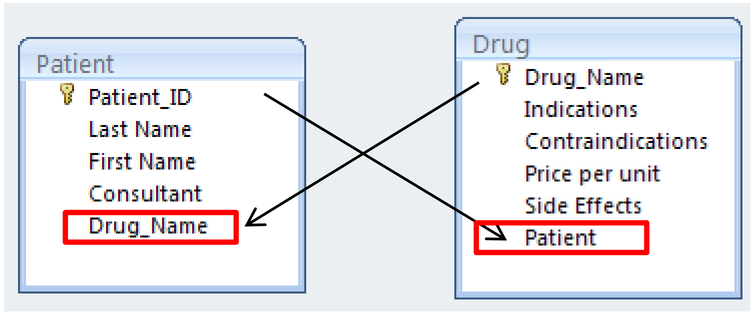
Patient

Patient	Last Name	First Name	Consultant
1	Bedecs	Anna	JBS_1955
2	Gratacos Solso	Antonio	JBS_1955
3	Axen	Thomas	JBS_1955
4	Lee	Christina	HSG_1998
5	O'Donnell	Martin	JBS_1955
6	Pérez-Olaeta	Francisco	HSG_1998

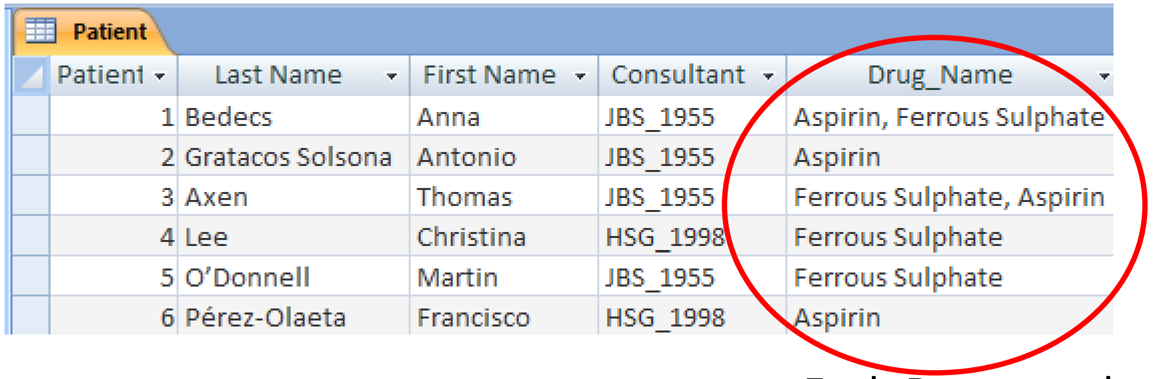
Each Consultant appears once in Consultant but many times in Patient

Many to Many (m:n)





Each Patient may be prescribed many Drugs

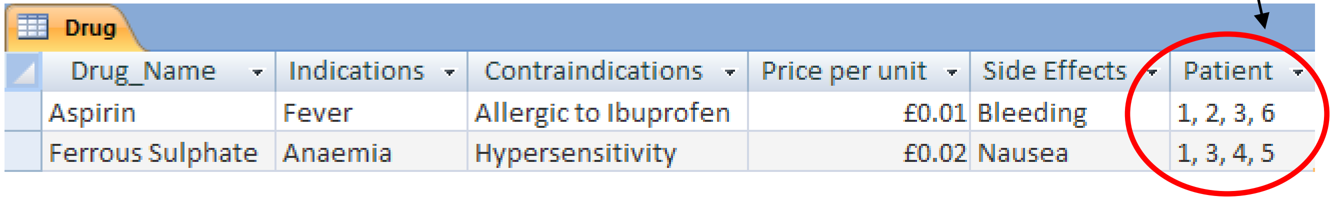


Patient

Patient	Last Name	First Name	Consultant	Drug_Name
1	Bedecs	Anna	JBS_1955	Aspirin, Ferrous Sulphate
2	Gratacos Solsona	Antonio	JBS_1955	Aspirin
3	Axen	Thomas	JBS_1955	Ferrous Sulphate, Aspirin
4	Lee	Christina	HSG_1998	Ferrous Sulphate
5	O'Donnell	Martin	JBS_1955	Ferrous Sulphate
6	Pérez-Olaeta	Francisco	HSG_1998	Aspirin

BAD design:
Consider
insert/update/delete
transactions

Each Drug may be prescribed to many Patients



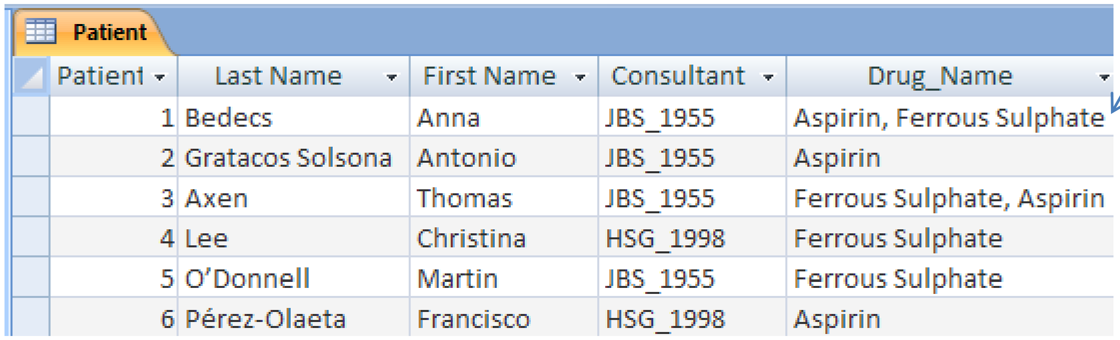
Drug

Drug_Name	Indications	Contraindications	Price per unit	Side Effects	Patient
Aspirin	Fever	Allergic to Ibuprofen	£0.01	Bleeding	1, 2, 3, 6
Ferrous Sulphate	Anaemia	Hypersensitivity	£0.02	Nausea	1, 3, 4, 5

Many to Many (m:n)

Always de-compose **m:n** relationships into **1:n** relationships because:

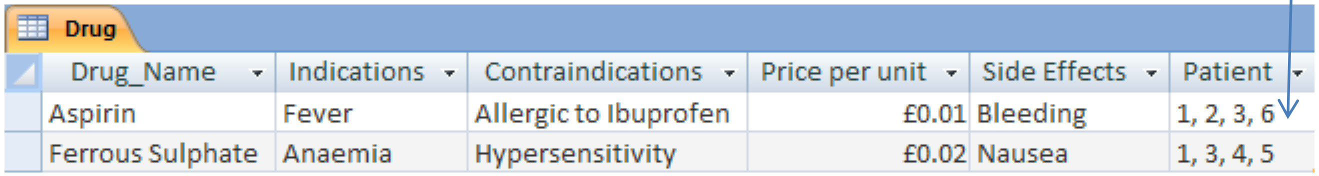
It is impossible to enforce referential integrity on m:n joins
They can cause duplicated data (redundancy)
Insert, update and delete operations are very tricky!
They can cause more than one data item in each field – called a **repeating group**



Patient

Patient	Last Name	First Name	Consultant	Drug_Name
1	Bedecs	Anna	JBS_1955	Aspirin, Ferrous Sulphate
2	Gratacos Solsona	Antonio	JBS_1955	Aspirin
3	Axen	Thomas	JBS_1955	Ferrous Sulphate, Aspirin
4	Lee	Christina	HSG_1998	Ferrous Sulphate
5	O'Donnell	Martin	JBS_1955	Ferrous Sulphate
6	Pérez-Olaeta	Francisco	HSG_1998	Aspirin

Patient 1 is prescribed 2 drugs



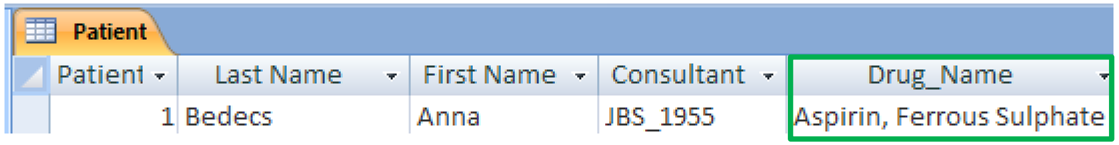
Drug

Drug_Name	Indications	Contraindications	Price per unit	Side Effects	Patient
Aspirin	Fever	Allergic to Ibuprofen	£0.01	Bleeding	1, 2, 3, 6
Ferrous Sulphate	Anaemia	Hypersensitivity	£0.02	Nausea	1, 3, 4, 5

Aspirin is prescribed to Patients 1, 2, 3, 6

An example of data redundancy:

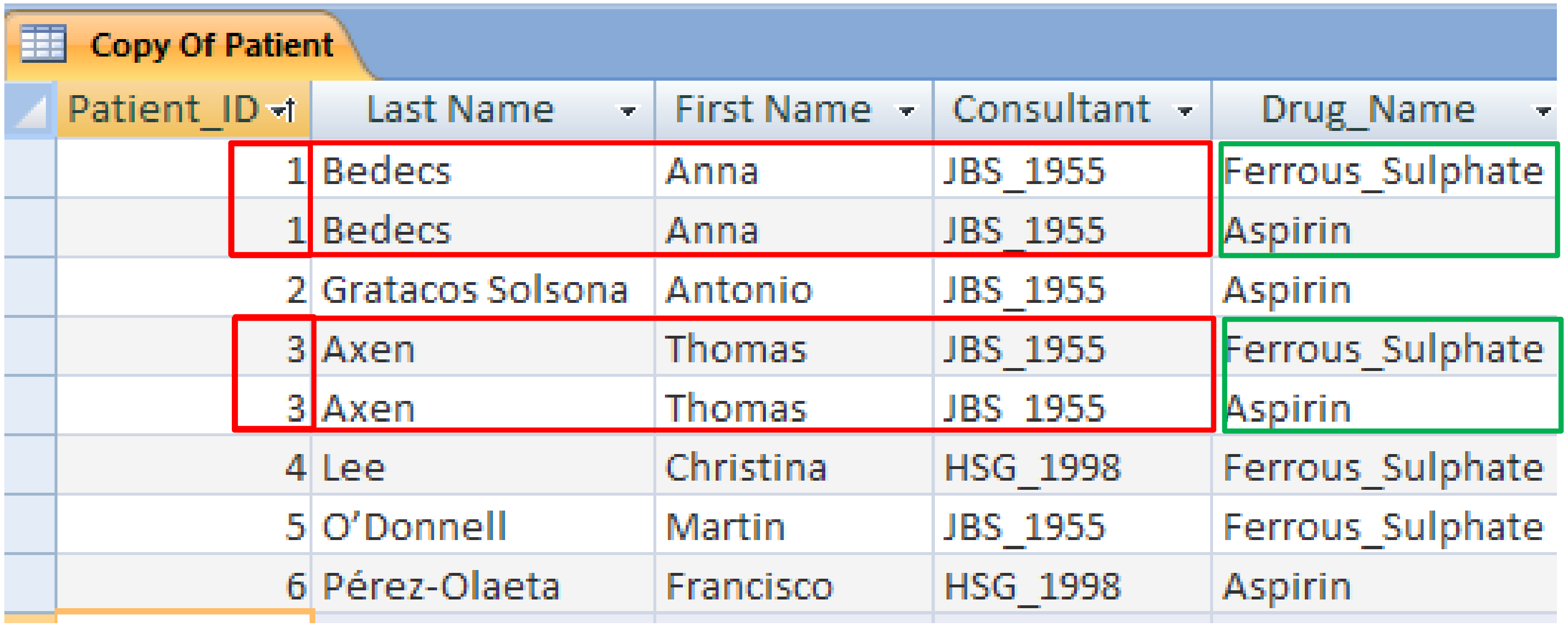
The **repeating group** has been resolved but at a cost!



Patient

Patient	Last Name	First Name	Consultant	Drug_Name
1	Bedecs	Anna	JBS_1955	Aspirin, Ferrous Sulphate

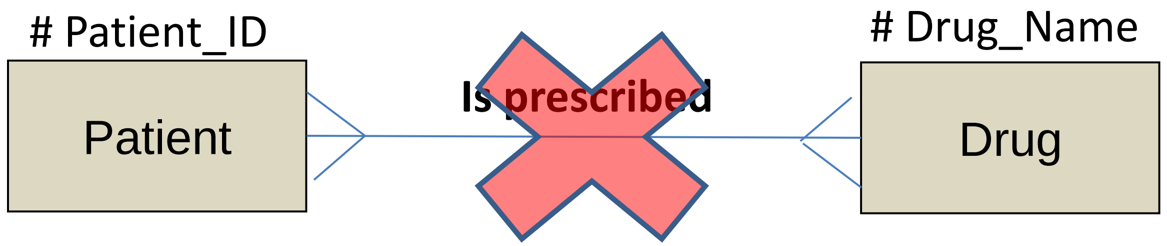
Also consider implications for the PK

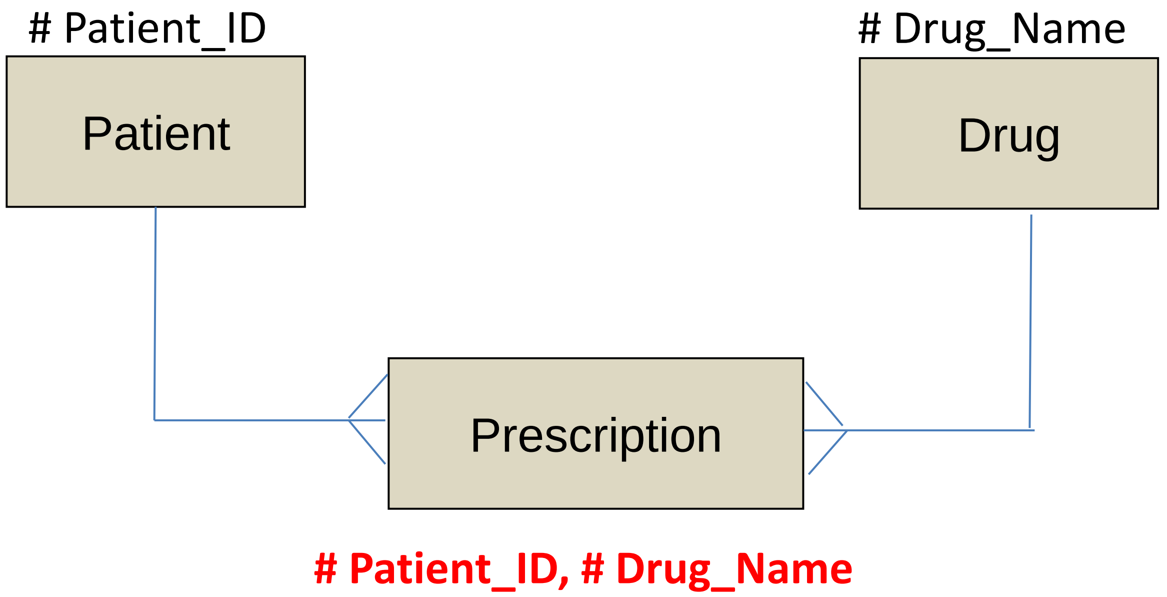


Copy Of Patient

Patient_ID	Last Name	First Name	Consultant	Drug_Name
1	Bedecs	Anna	JBS_1955	Ferrous_Sulphate
1	Bedecs	Anna	JBS_1955	Aspirin
2	Gratacos Solsona	Antonio	JBS_1955	Aspirin
3	Axen	Thomas	JBS_1955	Ferrous_Sulphate
3	Axen	Thomas	JBS_1955	Aspirin
4	Lee	Christina	HSG_1998	Ferrous_Sulphate
5	O'Donnell	Martin	JBS_1955	Ferrous_Sulphate
6	Pérez-Olaeta	Francisco	HSG_1998	Aspirin

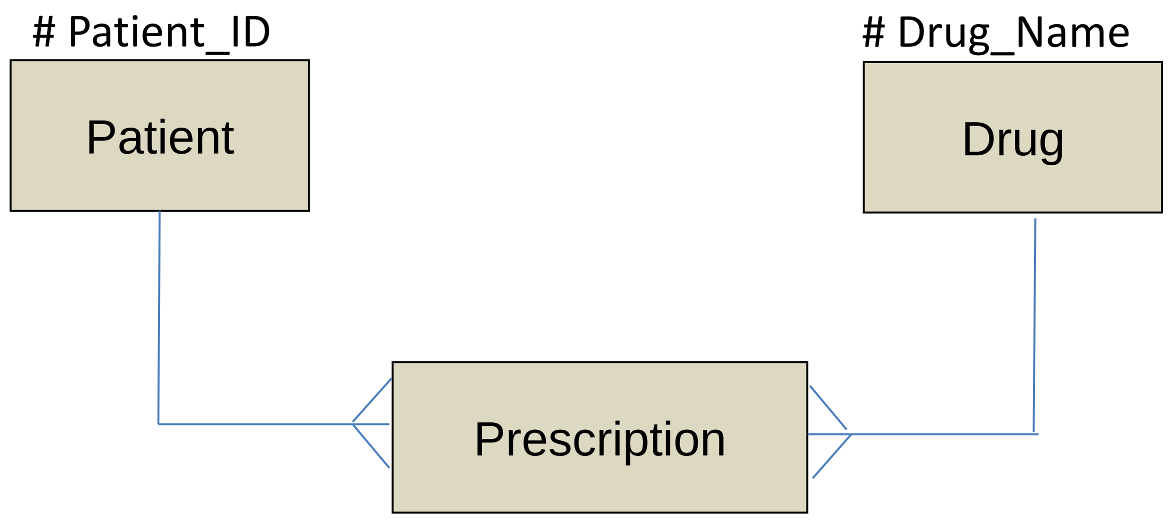
A simple rule for de-composing Many to Many Joins





Compound Keys are unique PKs using more than one field

A simple rule for de-composing Many to Many joins



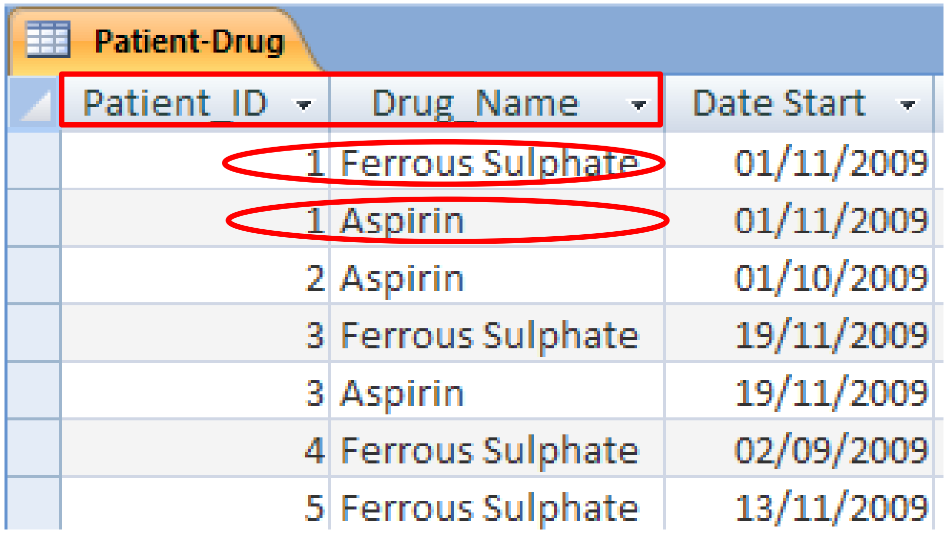
# Prescription_ID
FK Patient_ID
FK Drug_Name

A simple ID number is an alternative for the PK of Prescription
Now Patient_ID and Drug_Name are FKs

Task: Use the worksheet to de-compose the m:n joins
Illustrate both methods

Compound Keys work like this . . .

# Patient_ID, # Drug_Name



Patient-Drug

Patient_ID	Drug_Name	Date Start
1	Ferrous Sulphate	01/11/2009
1	Aspirin	01/11/2009
2	Aspirin	01/10/2009
3	Ferrous Sulphate	19/11/2009
3	Aspirin	19/11/2009
4	Ferrous Sulphate	02/09/2009
5	Ferrous Sulphate	13/11/2009

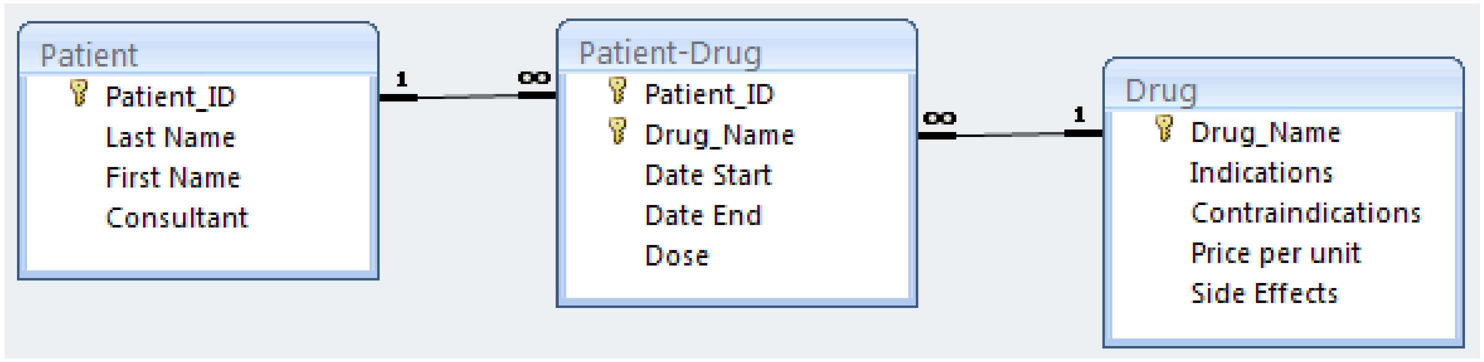
The values of Patient_ID ‘1’ and Drug_Name ‘Ferrous Sulphate’ cannot be entered in another row in this table

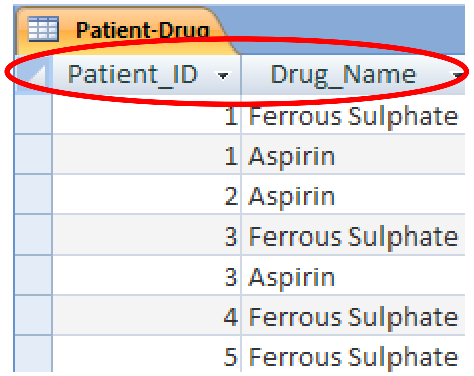
A patient has a row for each drug they are prescribed

When implemented, the resolved m:n looks like this . . .

Each Patient
may be prescribed
many drugs

Each Drug
may be prescribed
to many patients





Patient-Drug

Patient_ID	Drug_Name
1	Ferrous Sulphate
1	Aspirin
2	Aspirin
3	Ferrous Sulphate
3	Aspirin
4	Ferrous Sulphate
5	Ferrous Sulphate

Each row has a unique compound PK

Normalisation

1st Normal Form:
Each field must hold only **one piece of data** (relevant to the PK)

2nd Normal Form:
Each non PK field is relevant to the **whole PK** (when PK is compound)

3rd normal Form:
Check for potential PKs in the non PK fields (**avoid dependencies**)

Dependencies

STORE

# Store ID
Name
Address Line
City
County
Post code
Country
Phone number
Comments

If I know the # StoreID then I know the Name, Address Line, City, County, Post Code, Country, Phone no, Comments

Name is dependent on Store ID

Address line is dependent on Store ID

City is dependent on Store ID

and so on . . .

Remember:
1NF: Each non key field must be one data item related (i.e. 'dependent') on the PK

1st Normal Form:

Each field must hold only **one piece of data** (relevant to the PK)

Table **design** for the business rule ‘Stores are our customers’

STORE
Store ID
Name
Address
Phone number
Comments

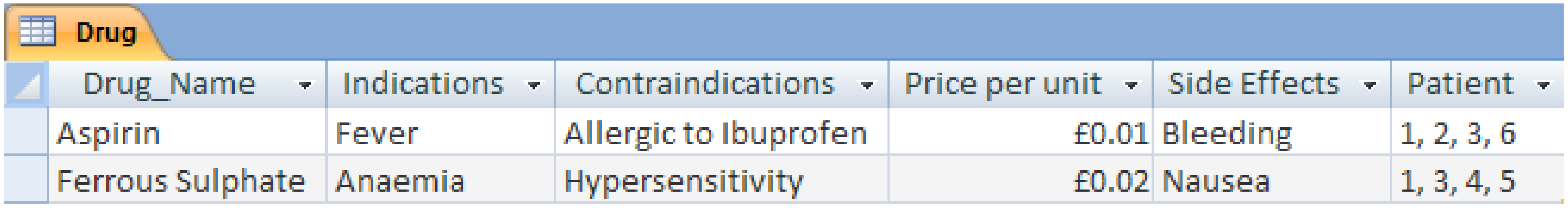
Not good

‘Address’ holds more than one piece of data

STORE
Store ID
Name
Address Line
City
County
Post code
Country
Phone number
Comments

Good

Another example of the violation of 1NF



Drug

Drug_Name	Indications	Contraindications	Price per unit	Side Effects	Patient
Aspirin	Fever	Allergic to Ibuprofen	£0.01	Bleeding	1, 2, 3, 6
Ferrous Sulphate	Anaemia	Hypersensitivity	£0.02	Nausea	1, 3, 4, 5

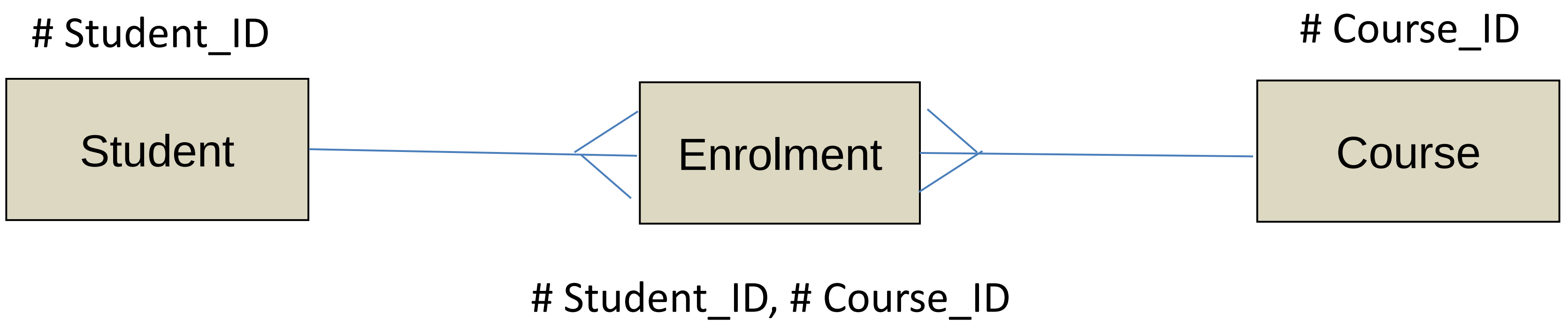
This is called a ‘repeating group’
The cell is storing more than one fact

1st Normal Form:
Each field must hold only **one piece of data** (relevant to the PK)

2nd Normal Form:

Each non PK field is relevant to the **whole PK** (when PK is compound)

Only consider 2NF when you have a **compound** key



Enrolment(# Student_ID, # Course_ID, DateOfEnrolment, StudentName, CourseName)

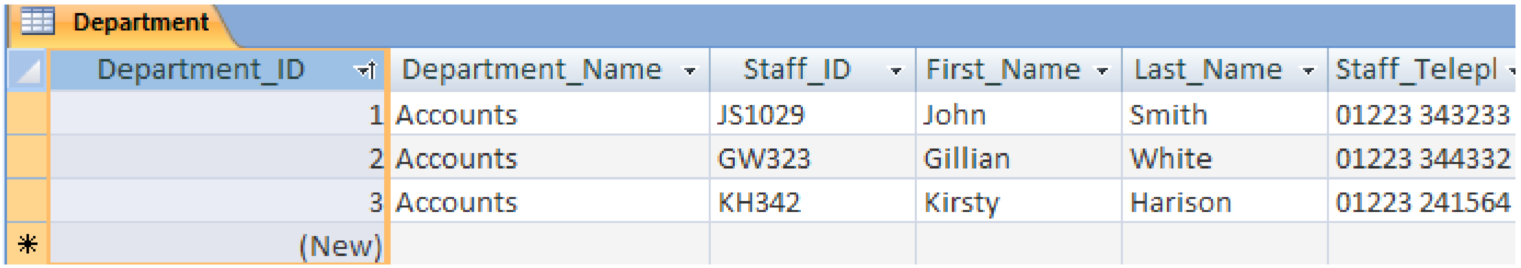
Which non key fields are NOT fully dependent on the compound key?

Third Normal Form

3NF: Check for potential PKs in the non PK fields (**avoid dependencies**)

DEPARTMENT
# Department ID
Department_Name
Staff_ID
First_Name
Last_Name
Staff_Telephone

Suppose we wanted to track **Staff** who work in **Departments** . . .



Department

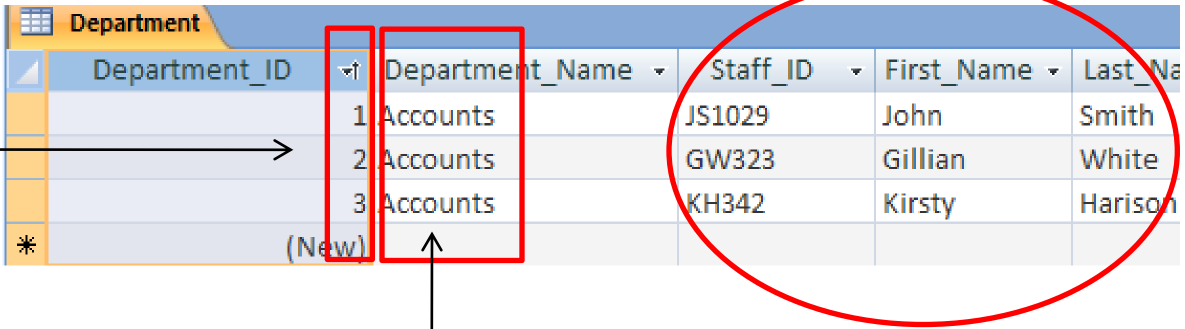
Department_ID	Department_Name	Staff_ID	First_Name	Last_Name	Staff_Telepl
1	Accounts	JS1029	John	Smith	01223 343233
2	Accounts	GW323	Gillian	White	01223 344332
3	Accounts	KH342	Kirsty	Harison	01223 241564
(New)					

Can you see any problems?

Violation of 3NF

The PK is identifying each row uniquely (good) but is not identifying each Department uniquely (bad).

Each **staff** entry is forcing the data onto many rows.



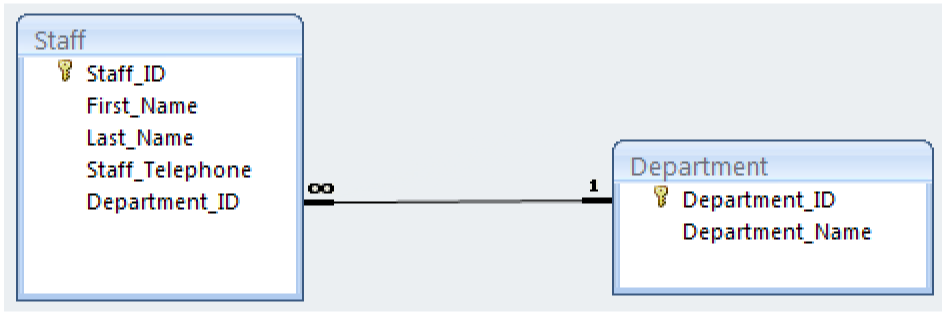
Department_ID	Department_Name	Staff_ID	First_Name	Last_Na
1	Accounts	JS1029	John	Smith
2	Accounts	GW323	Gillian	White
3	Accounts	KH342	Kirsty	Harison
(New)				

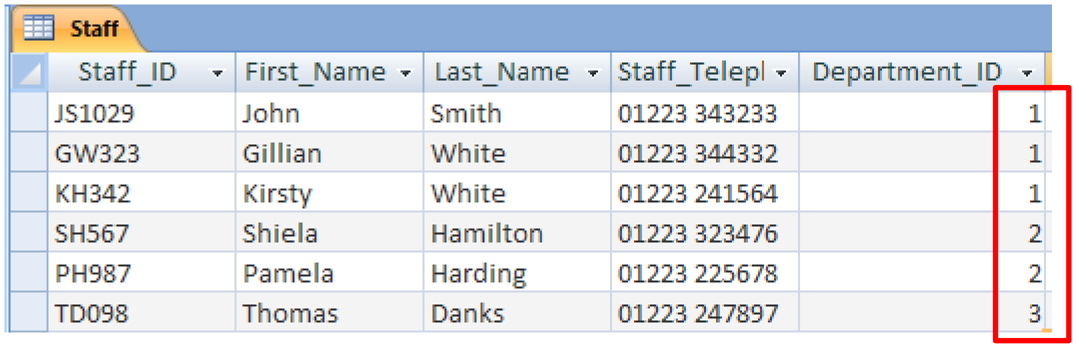
Data redundancy is occurring (bad)

Tables that are not in 3NF cause insert, update and delete anomalies:

Insert: What if a new department is created but no staff are allocated?
Update: What if the accounts department changed its Department_ID?
Delete: What if a department was deleted?

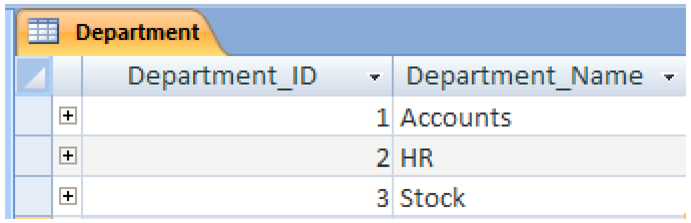
To resolve into correct 3NF, split the table
Leave a copy of the Department_ID in Staff to make the join





Staff_ID	First_Name	Last_Name	Staff_Telepl	Department_ID
JS1029	John	Smith	01223 343233	1
GW323	Gillian	White	01223 344332	1
KH342	Kirsty	White	01223 241564	1
SH567	Shiela	Hamilton	01223 323476	2
PH987	Pamela	Harding	01223 225678	2
TD098	Thomas	Danks	01223 247897	3

Many Staff to each Department


Department

Department_ID	Department_Name
1	Accounts
2	HR
3	Stock

This may look like redundancy but it is not. This is a FK field and makes the link between Department and Staff tables.

A worked example

Developing an ERD from a business scenario

Step 1 – Identify the data
Step 2 – Group the data into tables and allocate a primary key for each table
Step 3 – Join the tables using PK to FK

Step 1 – Identify the data

Highlight the words that will be stored as data

From this narrative description identify the data to be stored :

“I’m a manager of the S**porting-Goods Wholesale Company** that operates worldwide to fill orders from retail sporting-goods stores. The stores are our customers. For each customer, we must track an ID and a name. We may track an address including the city, county, post code, country and phone number.

We need to record information about our stock including ID, description, price, amount in stock. We hold stock in warehouses to best fill the orders of our customers. For each order, we must track an ID. We track the date ordered, date shipped, and payment type when the information is available.

Each warehouse must have an ID for which we track an address including the city, county, post code, country and phone number.

Departments are responsible for placing and tracking the orders when our customers call. For each department, we must track the ID and name. We may also record general comments about a customer.”

Table or Field?

"I'm a manager of the **Sporting-Goods Wholesale Company** that operates worldwide to fill orders from retail sporting-goods stores. The **stores** are our customers. For each customer, we must track an **ID** and a **name**. We may track an **address** including the **city**, **county, post code**, **country** and **phone number.**

We need to record information about our **stock** including **ID**, **description**, **price**, **amount in stock**. We hold stock in **warehouses** to best fill the **orders** of our customers. For each order, we must track an **ID**. We track the **date ordered**, **date shipped**, and **payment type** when the information is available.

Each warehouse must have an **ID** for which we track an **address** including the **city**, **county, post code**, **country** and **phone number**.

Departments are responsible for placing and tracking the orders when our customers call. For each department, we must track the **ID** and **name**. We may also record general **comments** about a customer."

Step 2a – Group the fields into tables
ensure data is split into single data items

STORE
Store ID
Name
Address Line
City
County
Post code
Country
Phone number
Comments

ORDER
Order ID
Date ordered
Date shipped
Payment type

WAREHOUSE
Warehouse ID
Address Line
City
County
Post code
Country
Phone number

STOCK
Stock ID
Description
Price
Amount in stock

DEPARTMENT
Department ID
Name

Each field is listed once (unless involved in a PK to FK join)
and is in the correct table

Step 2b – allocate a PK for each table

STORE
- **# Store ID**
- Name
- Address
- City
- County
- Post code
- Country
- Phone number
- Comments

ORDER
- **# Order ID**
- Date ordered
- Date shipped
- Payment type

STOCK
- **# Stock ID**
- Description
- Price
- Amount in stock

WAREHOUSE
- **# Warehouse ID**
- Address
- City
- County
- Post code
- Country
- Phone number

DEPARTMENT
- **# Department ID**
- Name

Identifying relationships (joins)

find out the **rules** of the database – understand what you are trying to model

Step 3 – Join the tables using PK to FK

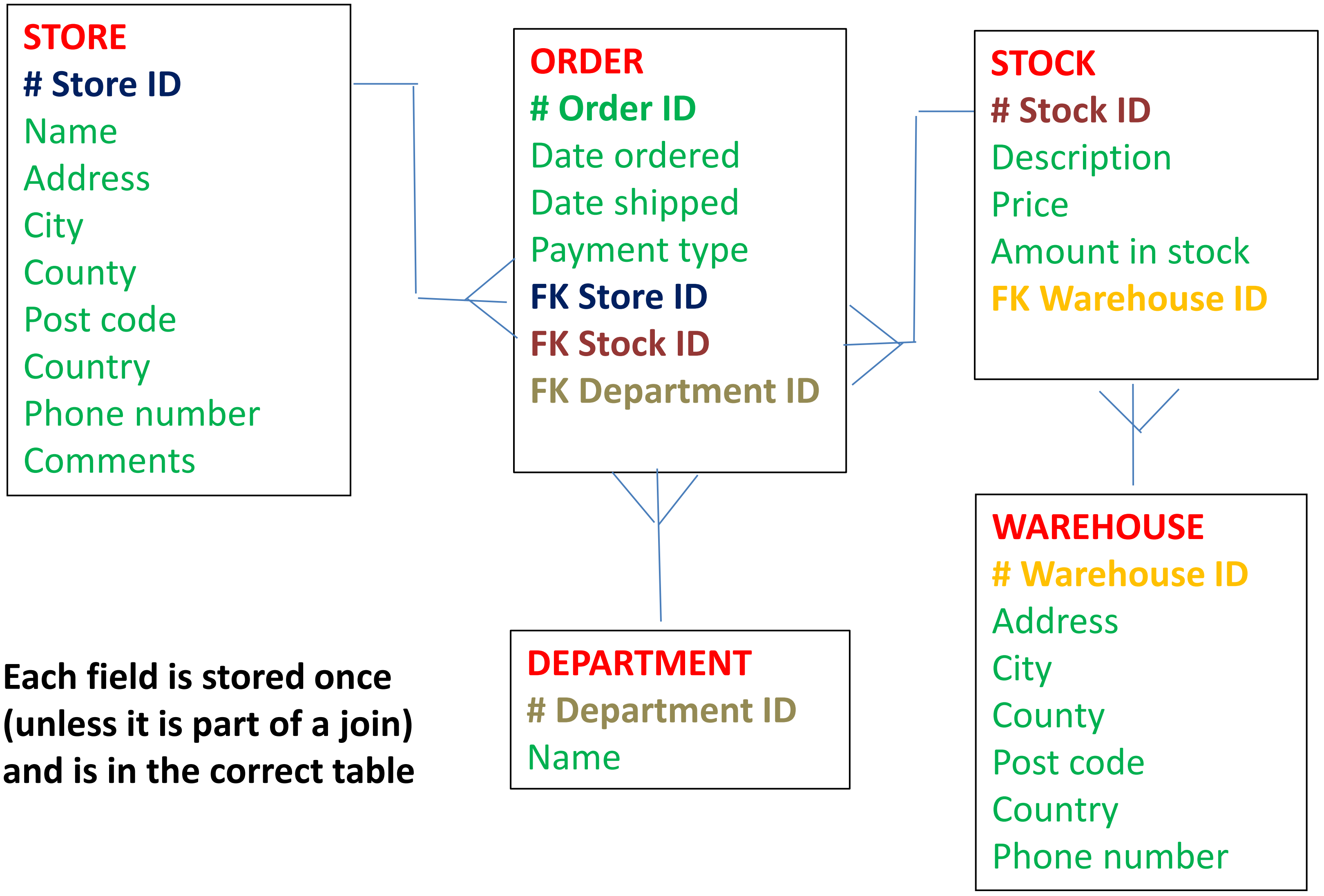
"I'm a manager of the **Sporting-Goods Wholesale Company** that operates worldwide to fill orders from retail sporting-goods stores. The stores are our customers. For each customer, we must track an ID and a name. We may track an address including the city, county, post code, country and phone number.

We need to record information about our stock including ID, description, price, amount in stock. **We hold stock in warehouses** to best fill the **orders of our customers**. For each order, we must track an ID. We track the date ordered, date shipped, and payment type when the information is available. **(Stock is in an order)**

Each warehouse must have an ID for which we track an address including the city, county, post code, country and phone number.

Departments are responsible for placing and tracking the **orders** when our customers call. For each department, we must track the ID and name. We may also record general comments about a customer."

Step 3 – Join the tables using PK to FK



Each field is stored once (unless it is part of a join) and is in the correct table

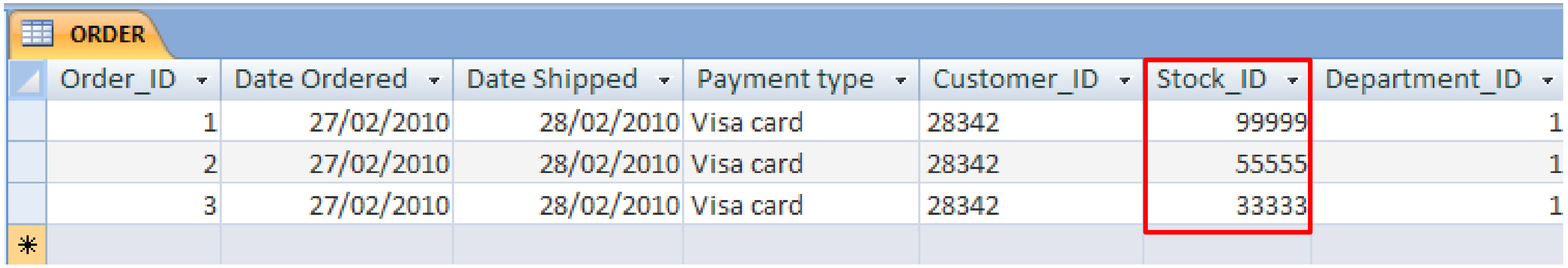
The ERD is not quite there, but OK for a 1st draft

ORDER
# Order ID
Date ordered
Date shipped
Payment type
FK Store ID
FK Stock ID
FK Department ID

This table needs further refinement, can you see why?

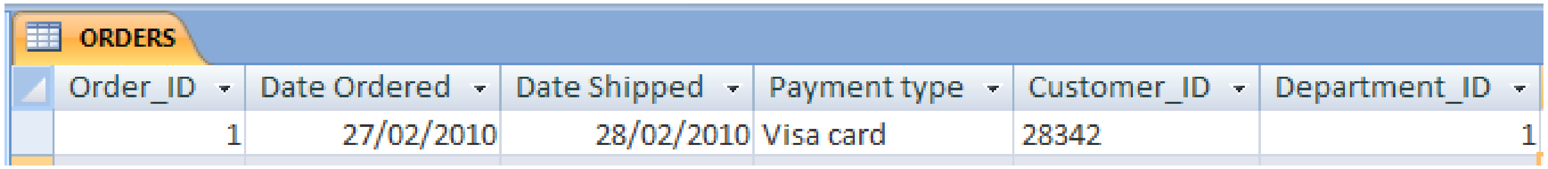
Task: Complete the occurrence table with dummy data.
Add three items into one order. Can you identify the problem?

Because the customer orders more than one stock item (repeating group) this is causing data redundancy.


ORDER

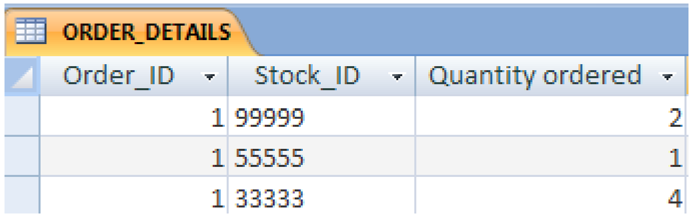
Order_ID	Date Ordered	Date Shipped	Payment type	Customer_ID	Stock_ID	Department_ID
1	27/02/2010	28/02/2010	Visa card	28342	99999	1
2	27/02/2010	28/02/2010	Visa card	28342	55555	1
3	27/02/2010	28/02/2010	Visa card	28342	33333	1
*						

Each stock item ordered requires a new row and a unique PK, the table is not in 1NF



ORDERS

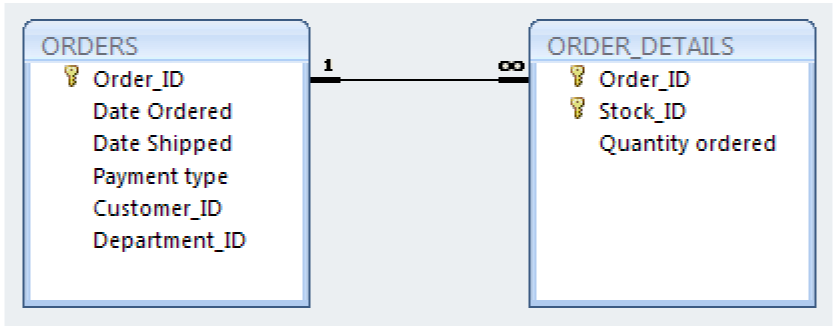
Order_ID	Date Ordered	Date Shipped	Payment type	Customer_ID	Department_ID
1	27/02/2010	28/02/2010	Visa card	28342	1

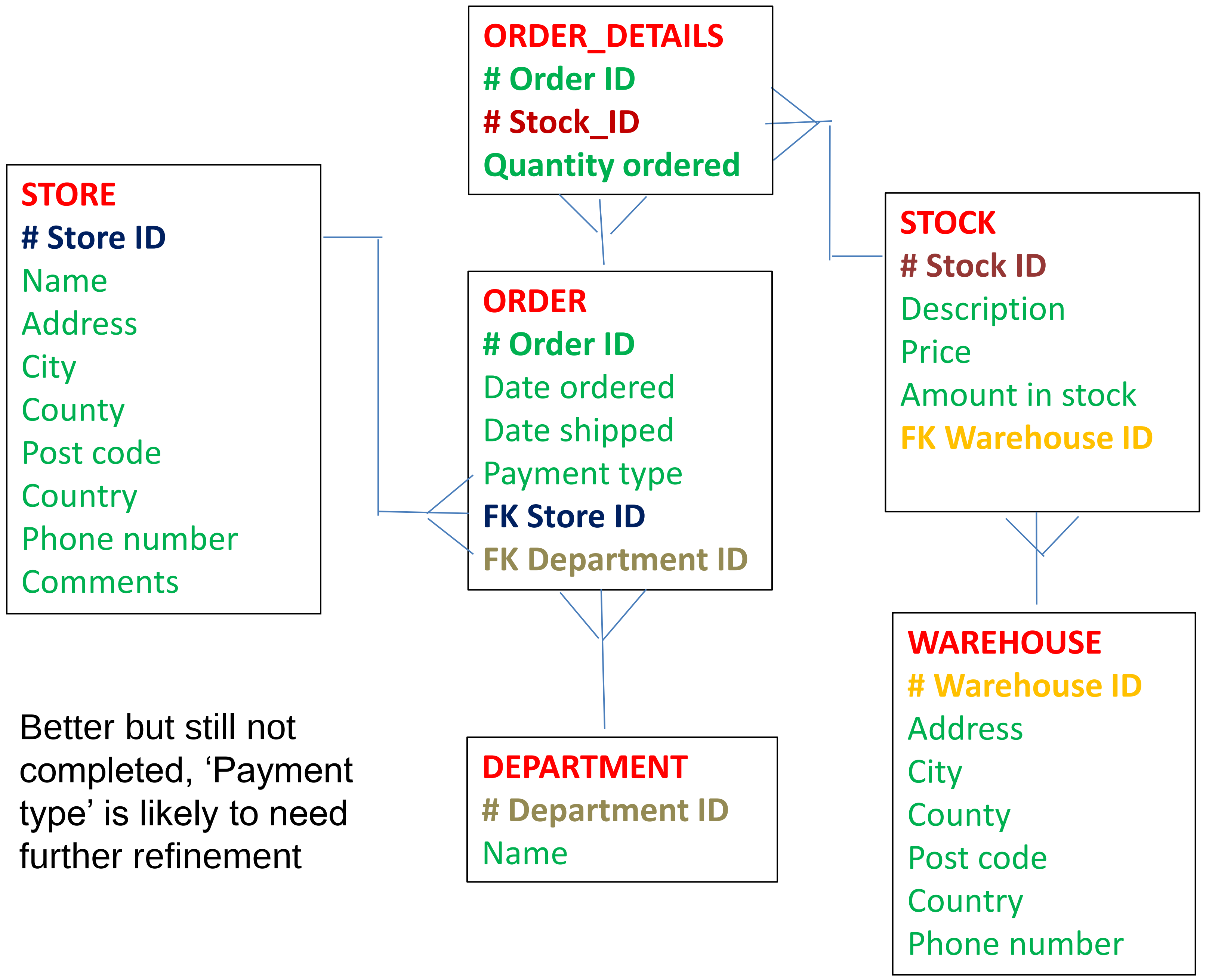


ORDER_DETAILS

Order_ID	Stock_ID	Quantity ordered
1	99999	2
1	55555	1
1	33333	4

Remove the Stock_ID field from the ORDER table
Place it into its own table
Take a copy of the ORDER_ID to make the join -
now a 1:n between ORDERS and ORDER_DETAILS
Adding ‘Quantity Ordered’ is a good idea too!





Better but still not completed, 'Payment type' is likely to need further refinement

ERD
Modelling Exercise

A horticultural research company is conducting a study of different plants grown using a new fertilizer. You are responsible for designing the database.

You need to store data on a variety of plants including scientific name, common name and family.

The environmental conditions of each greenhouse should be recorded including temperature and luminosity.

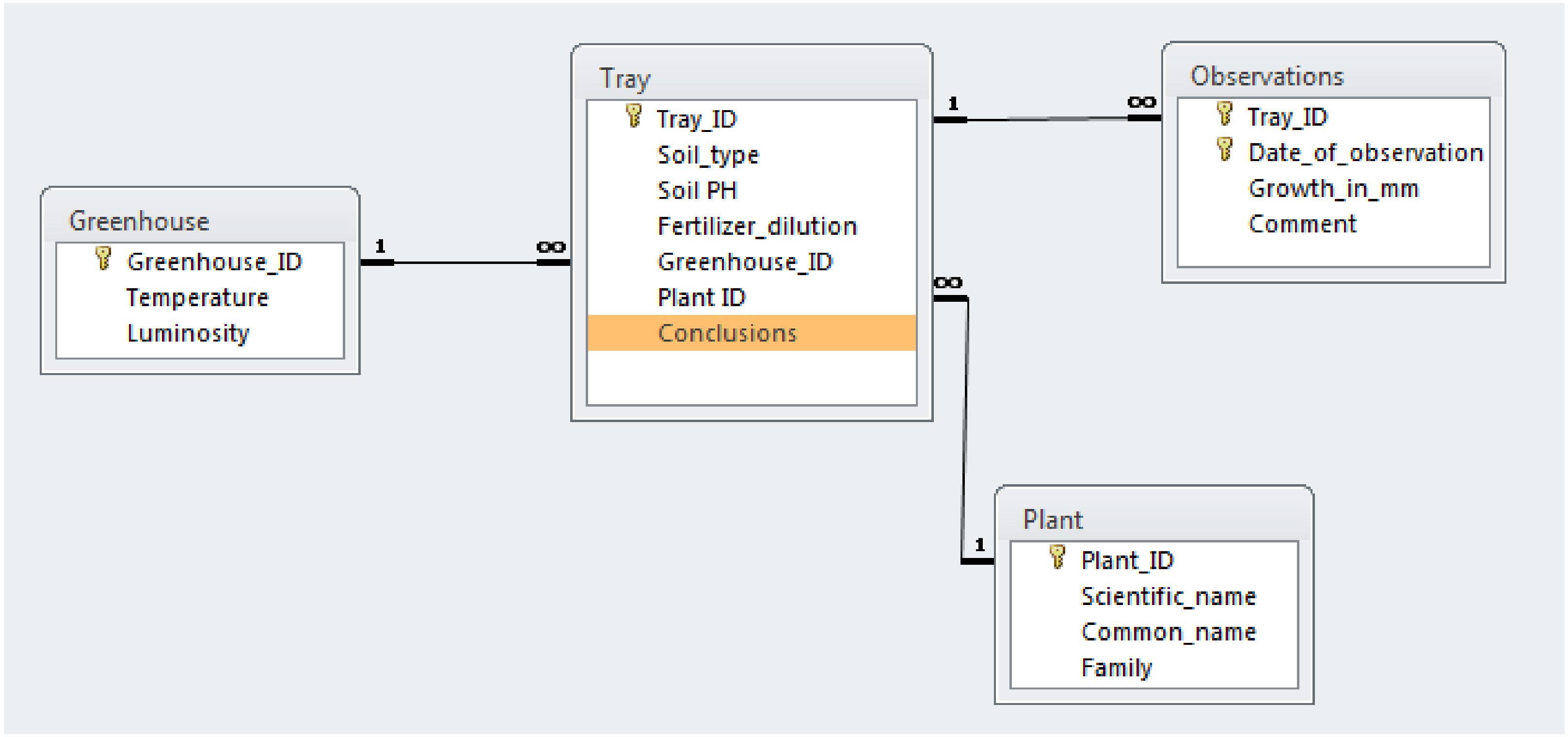
The soil substrate (type and PH) and the dilution of the new fertilizer is recorded for each tray.

A conclusion for each tray result should be recorded.

Daily observations of plant growth for each plant group must be stored with an optional comment.

Identify the data, and then model the entities in an ERD.

A possible solution



Each greenhouse has many trays with different soil conditions
Each plant may be grown in many different trays (with different conditions)
Each tray has many observations
Each conclusion is about an individual plant grown in a specific tray

Ear, Nose and Throat Research Study

Prior to **Flexible Laryngoscopy** topical nasal preparation is used to reduce patient discomfort and ease passage of the scope. The ideal nasal preparation should be comfortable for the patient, produce adequate anaesthesia and widen nasal patency.

Null hypothesis: There is no difference between co-phenylcaine spray or lidocaine /epinephrine nasal packing for preparing the nose prior to flexible laryngoscopy in terms of (1) patient comfort; and (2) degree of decongestion and ease of endoscope passage.

Patients rate experience on scale from 0 to 100 (zero = no pain) for each of these:

- Nasal Preparation (**NP**) bad taste, pain, anxiety and overall unpleasantness
- Flexible Laryngoscopy (**FL**) pain, anxiety, gagging and overall unpleasantness
- Patients have **one treatment** only

The surgeon used a visual analogue scale (VAS) from 0 -100 to record:

- Degree of decongestion
- Ease of endoscope passage

Other details recorded:

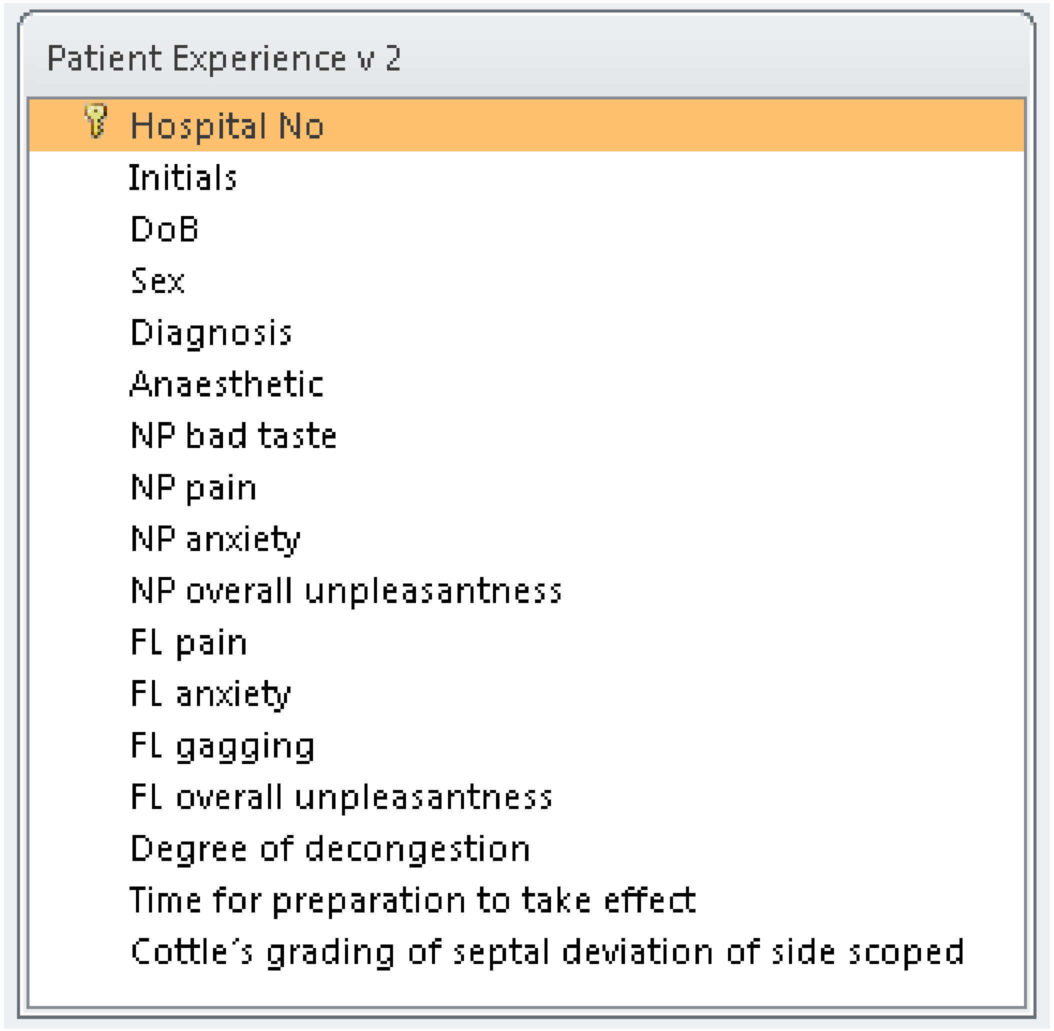
- Cottle's grading of septal deviation of side scoped
- Time given for preparation to take effect.

Microsoft Word 97 - 2003 Document

Microsoft Access Database

Adobe Acrobat Document

A possible solution



Patient Experience v 2

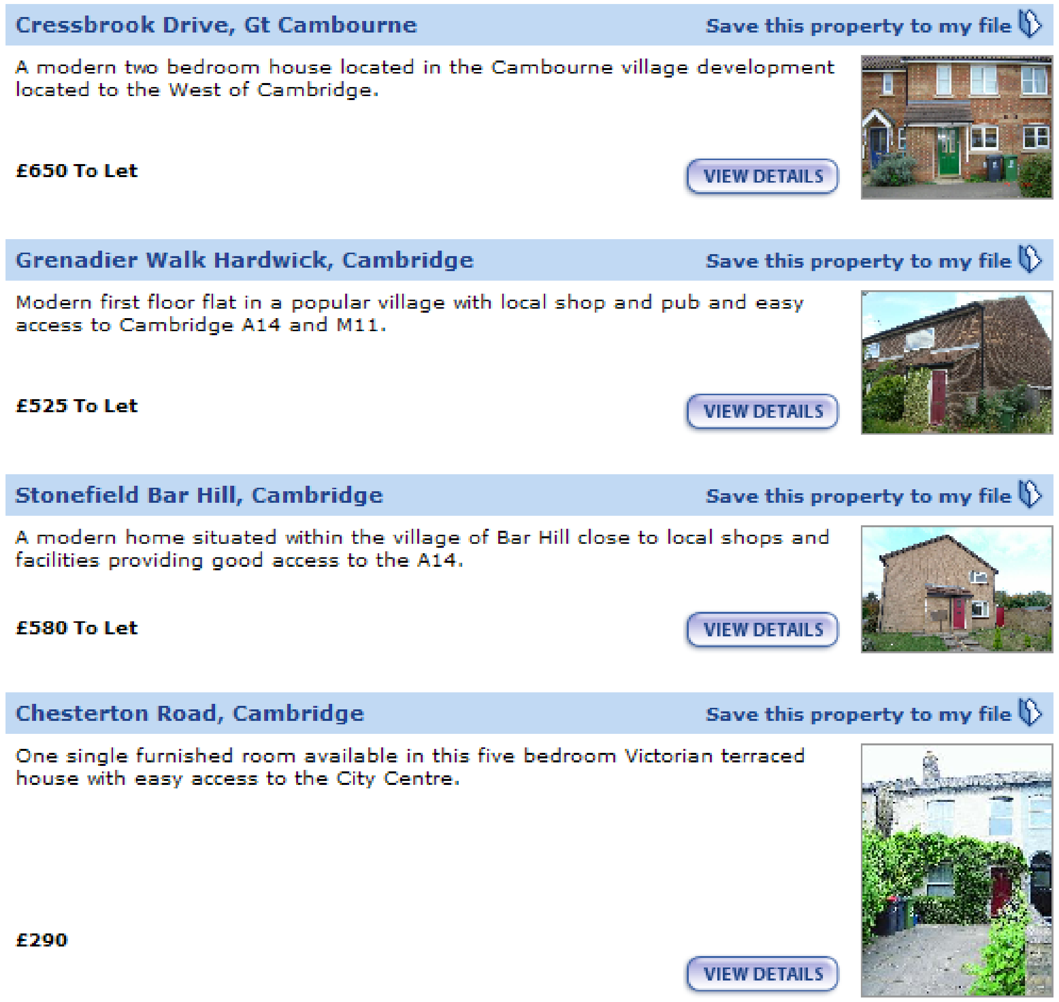
Hospital No
Initials
DoB
Sex
Diagnosis
Anaesthetic
NP bad taste
NP pain
NP anxiety
NP overall unpleasantness
FL pain
FL anxiety
FL gagging
FL overall unpleasantness
Degree of decongestion
Time for preparation to take effect
Cottle's grading of septal deviation of side scoped

This is easier than it first appears!

Each patient has one treatment using one anaesthetic and one set of observations.

A 'flat file' one table database could be used. A spread sheet package is an alternative application to use.

MyPlace Rental Properties



Cressbrook Drive, Gt Cambourne — Save this property to my file

A modern two bedroom house located in the Cambourne village development located to the West of Cambridge.

£650 To Let

Grenadier Walk Hardwick, Cambridge — Save this property to my file

Modern first floor flat in a popular village with local shop and pub and easy access to Cambridge A14 and M11.

£525 To Let

Stonefield Bar Hill, Cambridge — Save this property to my file

A modern home situated within the village of Bar Hill close to local shops and facilities providing good access to the A14.

£580 To Let

Chesterton Road, Cambridge — Save this property to my file

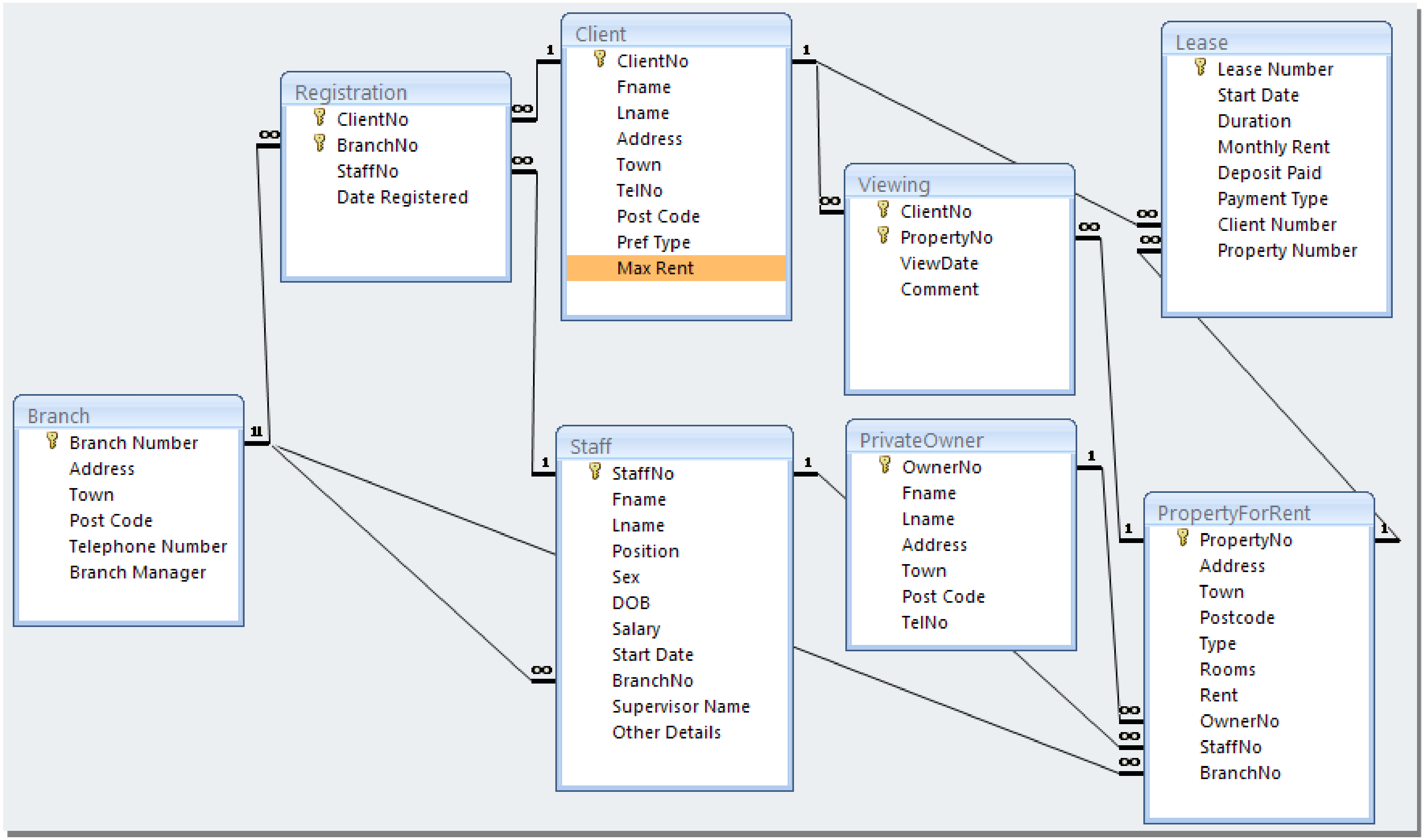
One single furnished room available in this five bedroom Victorian terraced house with easy access to the City Centre.

£290

Microsoft Office
rd 97 - 2003 Docum

MyPlace Estate Agents Case Study
A possible solution

Microsoft Access Database


Client
ClientNo
Fname
Lname
Address
Town
TelNo
Post Code
Pref Type
Max Rent
Registration
ClientNo
BranchNo
StaffNo
Date Registered
Lease
Lease Number
Start Date
Duration
Monthly Rent
Deposit Paid
Payment Type
Client Number
Property Number
Viewing
ClientNo
PropertyNo
ViewDate
Comment
Branch
Branch Number
Address
Town
Post Code
Telephone Number
Branch Manager
Staff
StaffNo
Fname
Lname
Position
Sex
DOB
Salary
Start Date
BranchNo
Supervisor Name
Other Details
PrivateOwner
OwnerNo
Fname
Lname
Address
Town
Post Code
TelNo
PropertyForRent
PropertyNo
Address
Town
Postcode
Type
Rooms
Rent
OwnerNo
StaffNo
BranchNo
1
∞