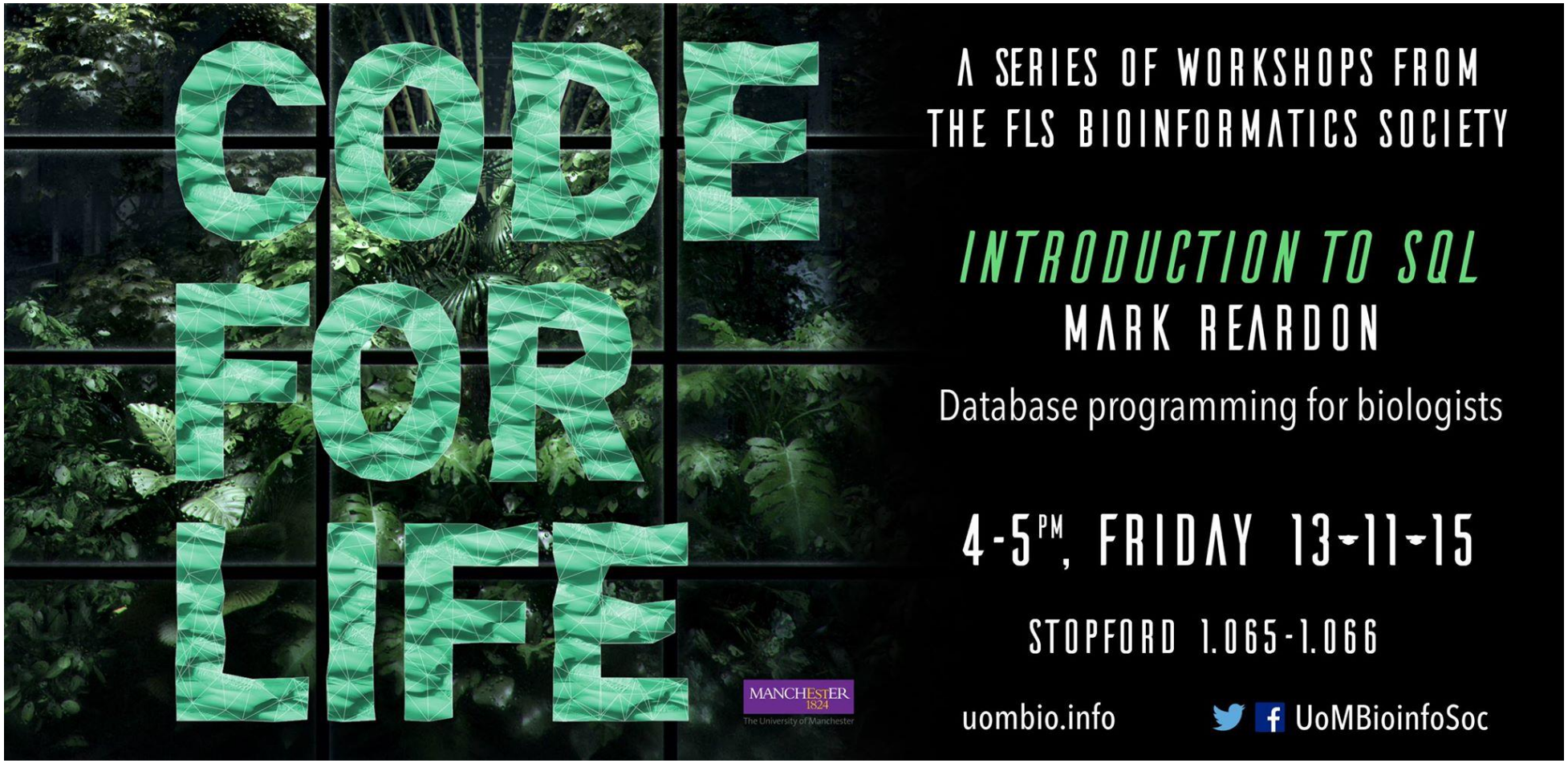
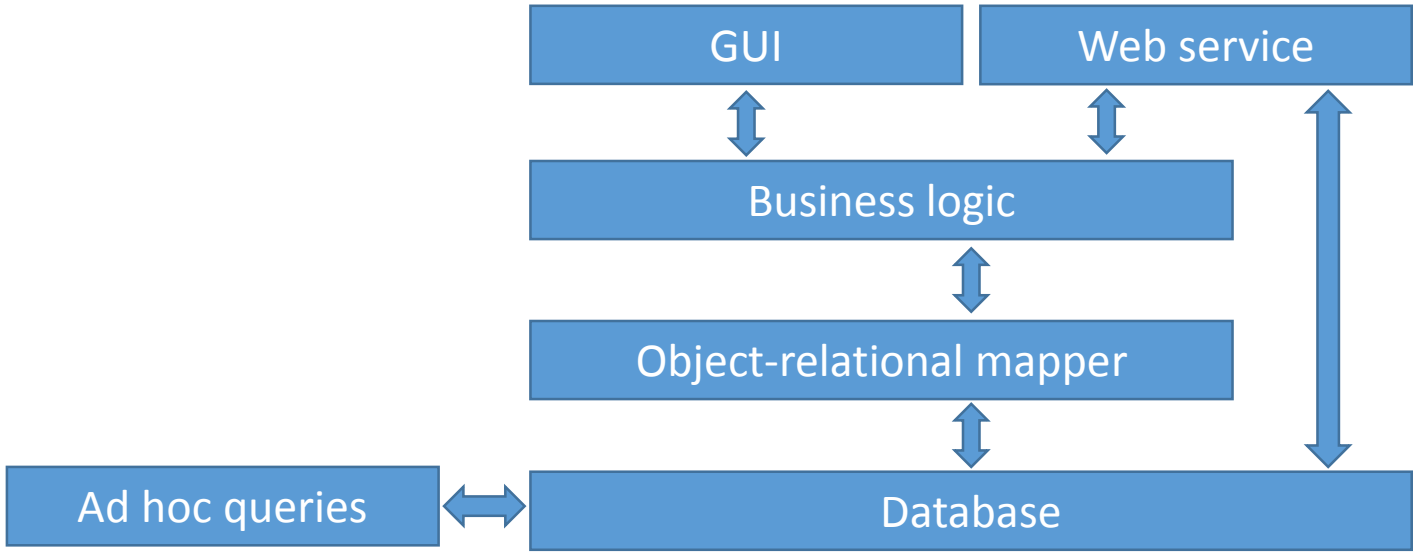

CODE FOR LIFE
MANCHESTER 1824
The University of Manchester
A SERIES OF WORKSHOPS FROM THE FLS BIOINFORMATICS SOCIETY
INTRODUCTION TO SQL
MARK REARDON
Database programming for biologists
4-5PM, FRIDAY 13-11-15
STOPFORD 1.065-1.066
uombio.info
UoMBioinfoSoc

Introduction

- Databases help us model the relationships between datasets
- Spreadsheets are not databases (please don't call them that!)
- There are many database engines
  - (MySQL, SQL Server, Oracle, Postgres, etc.)
- I'm good at SQL Server so we're using that
  - SQL Server Express is free and really quite powerful
- www.microsoft.com/en-us/server-cloud/products/sql-server-editions/sql-server-express.aspx
  - Just click 'Next' a few times and you'll probably be alright

Where do databases fit in?

- They can be standalone and used for ad hoc queries (this lecture)
- They can be a layer in a more complex application:

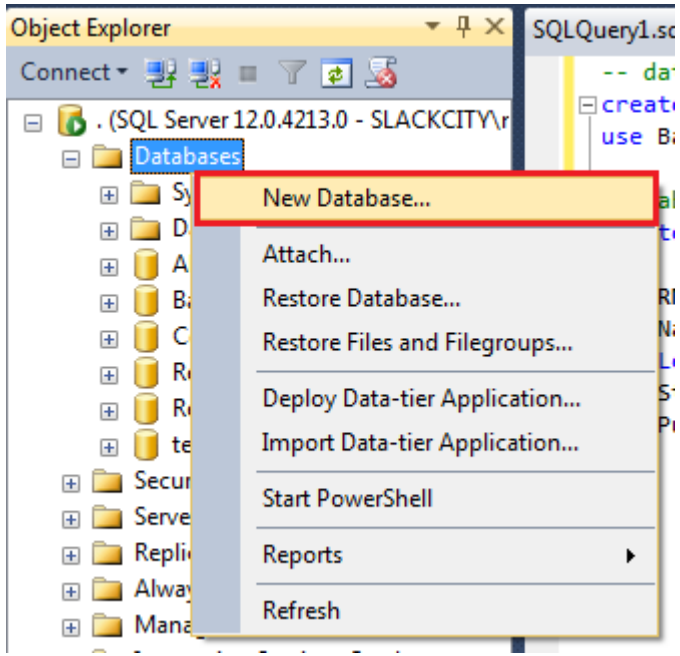


Creating a database

SQL

```sql
-- (re)create and use a database
if exists (select * from sysdatabases
           where Name = 'Basics')
       drop database Basics;
create database Basics;
use Basics;
```

Management Studio

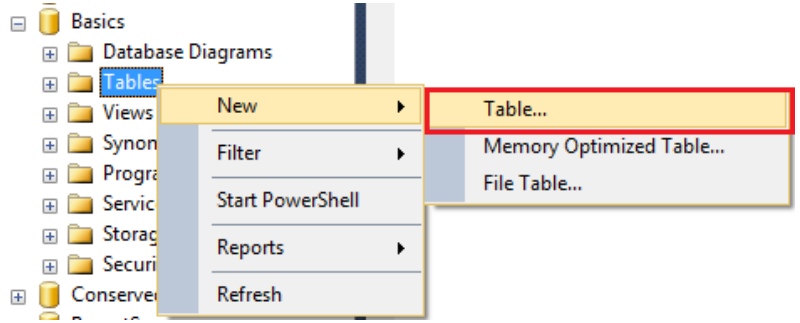


Creating a table to hold rows of data

SQL

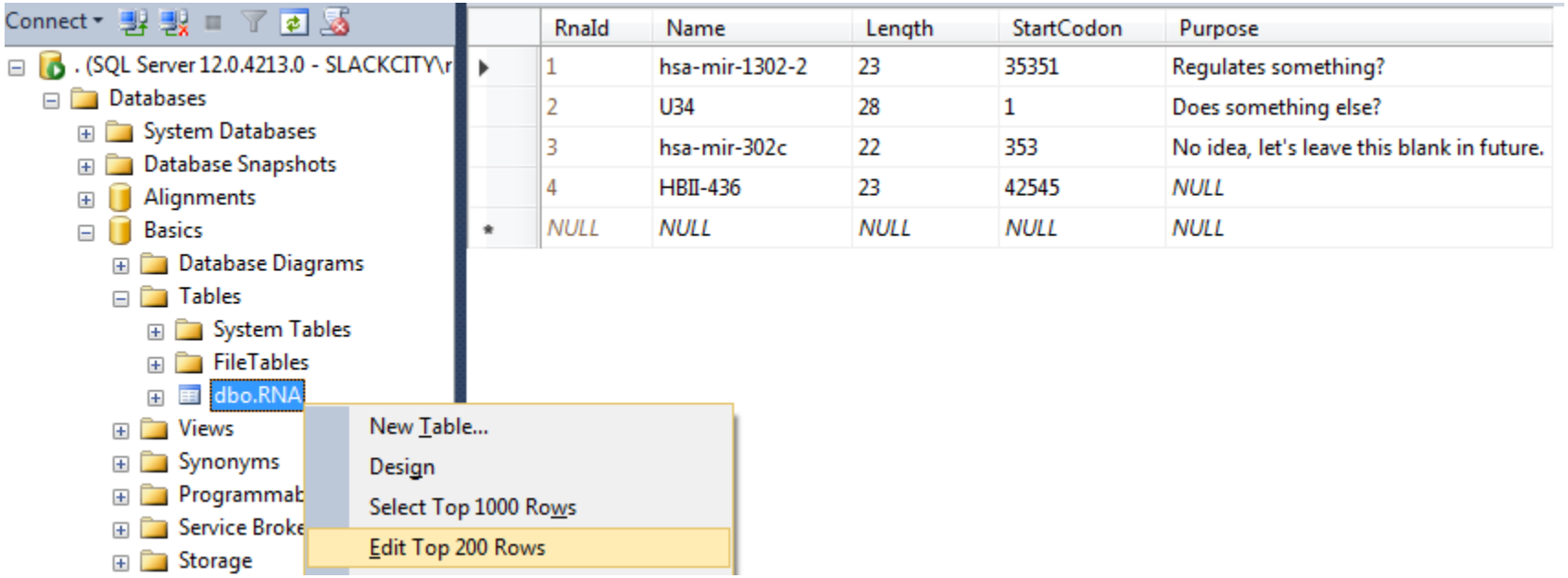
```sql
-- create a table
create table RNA
(
        RnaId int identity(1,1) primary key,
        Name varchar(50) not null,
        Length int not null,
        StartCodon int not null,
        Purpose varchar(255) null
);
```

Management Studio



Column Name	Data Type	Allow Nulls
RnaId	int	☐
Name	varchar(50)	☐
Length	int	☐
StartCodon	int	☐
Purpose	varchar(255)	☑

Table contents and CRUD



- Create, Retrieve, Update and Delete
- This covers pretty much everything you do to data

Table contents and CRUD

- Creating and retrieving data

```sql
-- create data
insert RNA (Name, Length, StartCodon, Purpose)
values ('hsa-mir-1302-2', 23, 35351, 'Regulates something?'),
       ('U34', 28, 1, 'Does something else?'),
       ('hsa-mir-302c', 22, 353, 'No idea, let''s leave this blank in future.'),
       ('HBII-436', 23, 42545, null);

-- retrieve data
select * from RNA where Name like 'hsa%';
```

100 %

Results | Messages

	RnaId	Name	Length	StartCodon	Purpose
1	1	hsa-mir-1302-2	23	35351	Regulates something?
2	3	hsa-mir-302c	22	353	No idea, let's leave this blank in future.

- Tip: select a part of the script and hit F5 to run that bit alone

Table contents and CRUD

- Updating data

```sql
-- update data
update RNA set Purpose = null where Name = 'U34';
select * from RNA where Name = 'U34';
```

100 %

Results | Messages

	Rnald	Name	Length	StartCodon	Purpose
1	2	U34	28	1	NULL

Table contents and CRUD

- Deleting data

```sql
-- delete data
delete from RNA where Name = 'U34';
select * from RNA;
```

100 %

Results | Messages

	RnaId	Name	Length	StartCodon	Purpose
1	1	hsa-mir-1302-2	23	35351	Regulates something?
2	3	hsa-mir-302c	22	353	No idea, let's leave this blank in future.
3	4	HBII-436	23	42545	NULL

Slicing and dicing

- Three statements and three result sets:

```sql
-- slicing and dicing data
select distinct Length from RNA;
select Length, count(*) as [Count] from RNA group by Length;
select * from RNA order by StartCodon;
```

100 %

Results | Messages

	Length
1	22
2	23

	Length	Count
1	22	1
2	23	2

	RnaId	Name	Length	StartCodon	Purpose
1	3	hsa-mir-302c	22	353	No idea, let's leave this blank in future.
2	1	hsa-mir-1302-2	23	35351	Regulates something?
3	4	HBII-436	23	42545	NULL

Primary keys

- Each record should have a way of uniquely identifying it

```
-- create a table
create table RNA
(
    RnaId int identity(1,1) primary key,
    Name varchar(50) not null,
    Length int not null,
    StartCodon int not null,
    Purpose varchar(255) null
);
```

- Records in other tables can be unambiguously linked to a record in this table (we'll see how in a moment)

Natural or arbitrary keys?

- You *could* use the RNA name as the key instead…
  - What if you want to change the name but it's been used elsewhere?
  - You'd have to find and change everywhere else the name has been used
  - You'd have to do this update everywhere *simultaneously…*

- Don't do it! Use auto-incrementing integers. Always.
  - Predictable Id mechanism
  - Field data can be edited without breaking links
  - The 'payload' fields can anyway be set to be unique if required
  - This argument was settled a long time ago

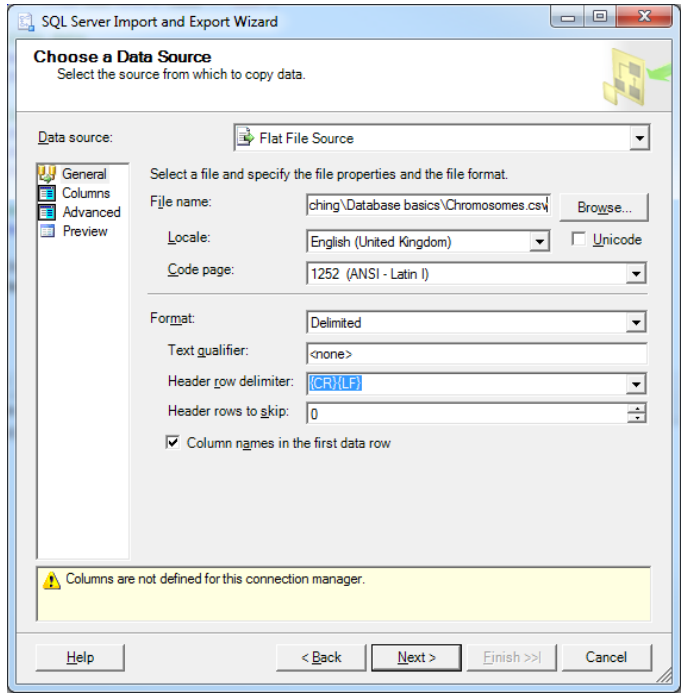
Foreign keys

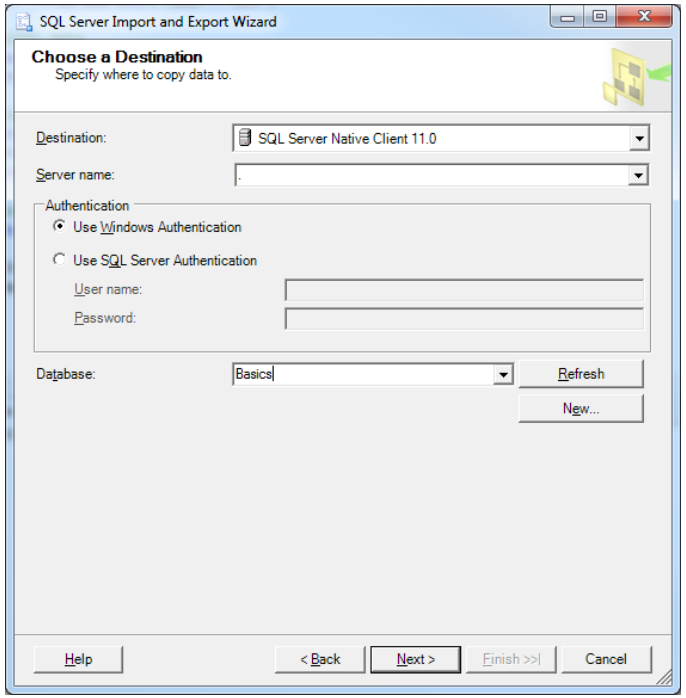
- A foreign key field is used to relate one table's data to another's
- Referential integrity: the database engine will enforce these links
  - This is one of the main advantages of relational databases
- But first, we need another table:

```
-- another table
create table Chromosomes
(
    ChromosomeId int identity(1,1) primary key,
    Name varchar(50)
);
```

Digression: Importing data

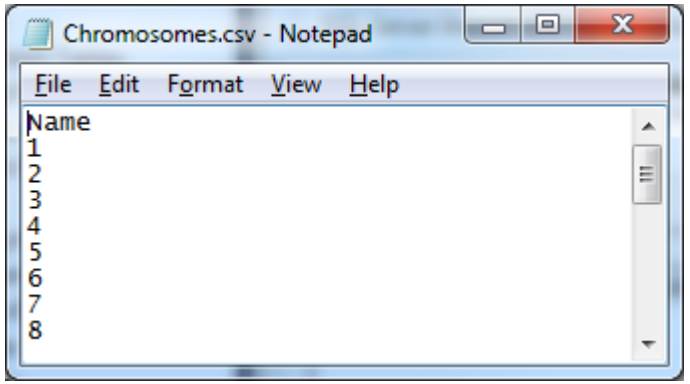
- Right-click on the database, choose Tasks, Import Data
- The source and destination matter most, just click ‘Next’ for the rest

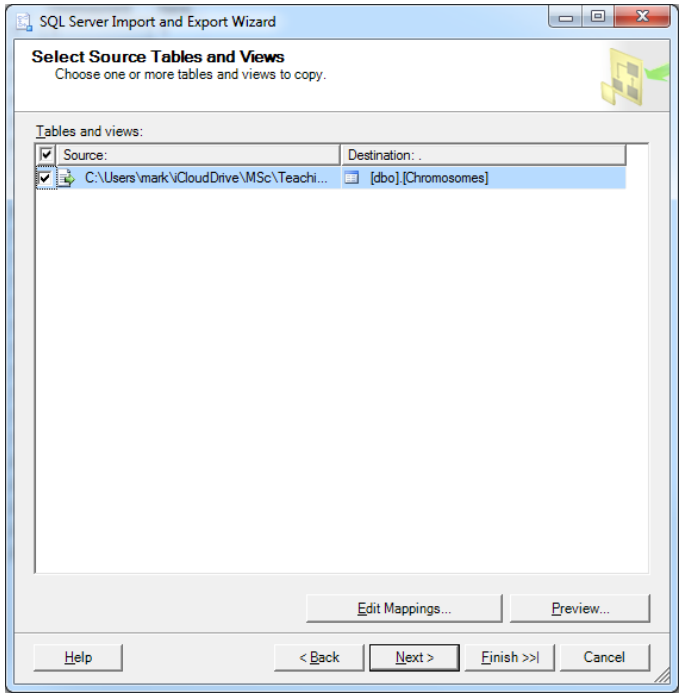




Digression: Still importing data

- SQL Server has spotted that we have a table with the right name
- It also maps to the right field because we named everything sensibly





Digression: Imported data

- The CSV has imported to the correct field as well
- SQL Server assigned Id values to each record

```
-- show the data after import
select * from Chromosomes;
```

.00 %

Results | Messages

	ChromosomeId	Name
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	X
24	24	Y

Foreign keys continued

- We need a link field in RNA to match the Id field in Chromosomes:

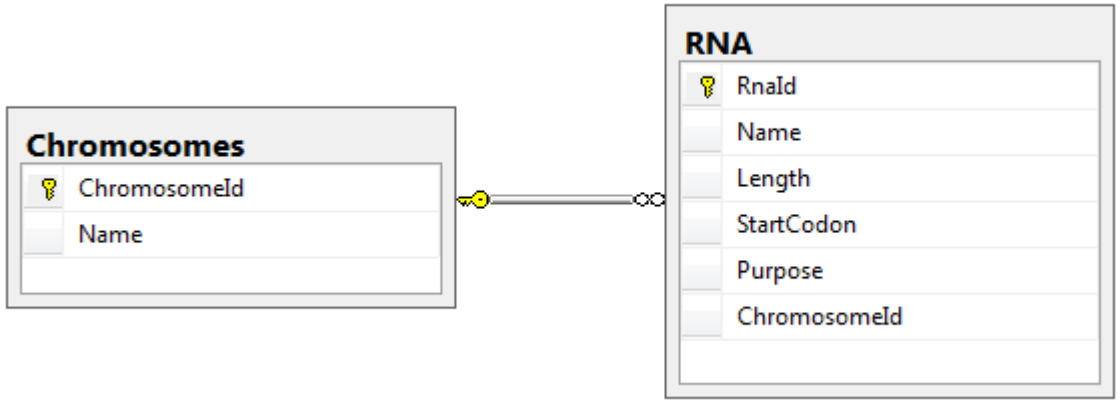
```
-- add a link field
alter table RNA add ChromosomeId int not null default(1);
```

- We relate the two tables with a ‘foreign key constraint’:

```
-- create the foreign key
alter table RNA add constraint FK_RNA_Chromosomes
     foreign key (ChromosomeId)
     references Chromosomes (ChromosomeId)
     on delete cascade;
```

Digression: database diagrams

- Right-click on Database Diagrams and New Database Diagram
- Select both tables and click Add and then Close



Foreign keys

- Why did we just do all this?
  - The RNAs are now all associated with a chromosome
  - RNAs *have* to have a chromosome (thanks to referential integrity)
  - Chromosomes *can* have associated RNAs (but they don't have to)
    - This is a one-to-many relationship

```
-- RNA chromosomes
select * from RNA;
```

100 %

Results | Messages

	RnaId	Name	Length	StartCodon	Purpose	ChromosomeId
1	1	hsa-mir-1302-2	23	35351	Regulates something?	1
2	3	hsa-mir-302c	22	353	No idea, let's leave this blank in future.	1
3	4	HBII-436	23	42545	NULL	1

Joins

- Join relate data from different tables together
- Takes into account the relationships between the tables
- There are various types:
  - Inner joins – results have values in both tables
  - Left outer joins – results have values in at least the left hand table
  - Cross joins, full outer joins, etc. – you'll hardly ever need these

Inner joins

- Combines records from two tables using a common field
- Only common values that are in both tables participate

```sql
-- chromosome RNAs
select c.Name as [Chromosome name], r.Name as [RNA name]
from   Chromosomes c
       inner join RNA r on c.ChromosomeId = r.ChromosomeId
```

100 %

Results | Messages

	Chromosome name	RNA name
1	1	hsa-mir-1302-2
2	1	hsa-mir-302c
3	1	HBII-436

```sql
-- chromosome RNA counts
select c.Name as [Chromosome name], count(*) as [RNA count]
from   Chromosomes c
       inner join RNA r on c.ChromosomeId = r.ChromosomeId
group by c.Name;
```

100 %

Results | Messages

	Chromosome name	RNA count
1	1	3

Left outer joins

- Combines records from two tables using a common field (like inner)
- Common values that are in left table are always present
- Missing values in right table are null

```sql
-- chromosome RNAs
select c.Name as [Chromosome name], r.Name as [RNA name]
from   Chromosomes c
       left outer join RNA r on c.ChromosomeId = r.ChromosomeId
```

100 %

Results | Messages

	Chromosome name	RNA name
1	1	hsa-mir-1302-2
2	1	hsa-mir-302c
3	1	HBII-436
4	2	NULL
5	3	NULL
6	4	NULL

Putting it all together

- Add some more RNA

```sql
-- add some more RNAs
insert RNA (Name, Length, StartCodon, Purpose, ChromosomeId)
values ('U10', 23, 565, null, 2),
       ('U11', 23, 565, null, 2),
       ('U12', 23, 565, null, 2),
       ('Xist', 22, 4453, 'Inactivates one X chromosome in females', 23),
       ('T', 23, 7987, null, 24);
select * from RNA;
```

100 %

Results | Messages

	RnaId	Name	Length	StartCodon	Purpose	ChromosomeId
1	1	hsa-mir-1302-2	23	35351	Regulates something?	1
2	3	hsa-mir-302c	22	353	No idea, let's leave this blank in future.	1
3	4	HBII-436	23	42545	NULL	1
4	5	U10	23	565	NULL	2
5	6	U11	23	565	NULL	2
6	7	U12	23	565	NULL	2
7	8	Xist	22	4453	Inactivates one X chromosome in females	23
8	9	T	23	7987	NULL	24

Putting it all together

- Chromosome RNA counting and length analysis

```sql
-- chromosome RNA counts
select c.Name as [Chromosome name], count(*) as [RNA count]
from   Chromosomes c
       inner join RNA r on c.ChromosomeId = r.ChromosomeId
group by c.Name;
```

100 %

Results | Messages

	Chromosome name	RNA count
1	1	3
2	2	3
3	X	1
4	Y	1

```sql
-- RNA length analysis
select Length, count(*) as [Count] from RNA group by Length;
```

100 %

Results | Messages

	Length	Count
1	22	2
2	23	6

Putting it all together

- RNA chromosome names

```sql
-- RNA chromosome names
select r.Name as [RNA name], c.Name as [Chromosome name]
from   Chromosomes c
       inner join RNA r on c.ChromosomeId = r.ChromosomeId
```

.00 %

Results | Messages

	RNA name	Chromosome name
1	hsa-mir-1302-2	1
2	hsa-mir-302c	1
3	HBII-436	1
4	U10	2
5	U11	2
6	U12	2
7	Xist	X
8	T	Y

Summary

- Create databases and tables
- CRUD
- Import data
- Create relationships between data
- Exploit those relationships to answer questions about the data
- Referential integrity prevents the data from becoming malformed

Next steps (next lecture?)

Indexes

- Primary keys are clustered indexes by default
- Foreign key fields should have indexes so joins are fast
- Fields that are part of queries should also have indexes

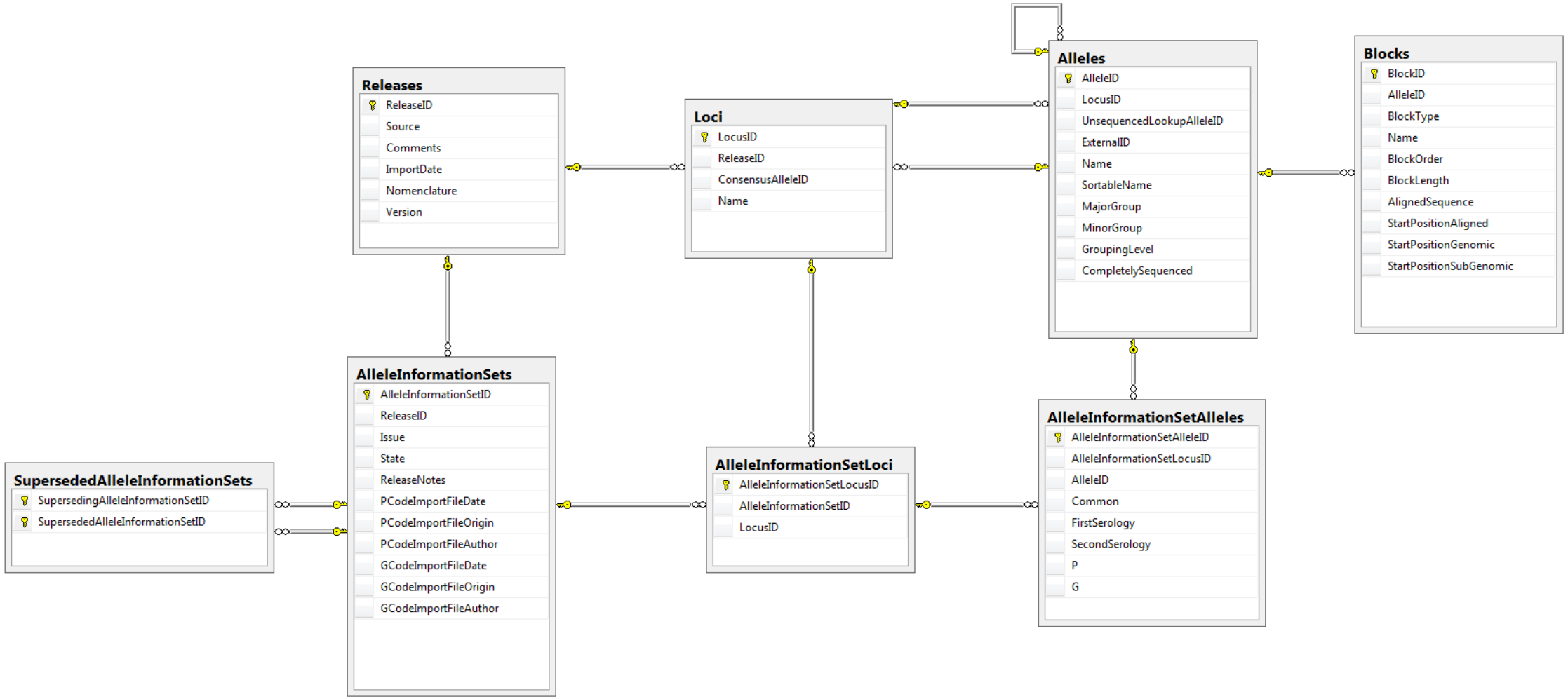
Stored procedures and functions

- Motivation
- Examples

Transactions

- Motivation
- Examples

Real world example 1



Real world example 2