Querying data with SQL

Introduction

Last week you learned some of the core theory of relational databases. This week you learn how to actually work with a relational database - in particular, how to extract data to answers questions of interest about that data.

To work with a relational database you typically use a **relational database management system** (or **RDBMS**). Popular ones are PostgreSQL, MySQL, and Oracle, but there are many others.

The RDBMS executes commands written in a language called "Structured Query Language", or "SQL" for short (often pronounced "seequell"). These commands tell the RDBMS to create a table, or insert some rows, or update some values, or search for certain results, and so on.

The RDBMS usually provides you with a graphical user interface, to help you construct SQL statements by clicking and dragging, without having to write any SQL code (you might not even see the code that gets constructed).

But these interfaces tend to have limits, and when you find yourself at a limit you will need to know some SQL. Even if you never encounter those limits it is still good to know some SQL - it will give you a solid understanding of how relational databases work.

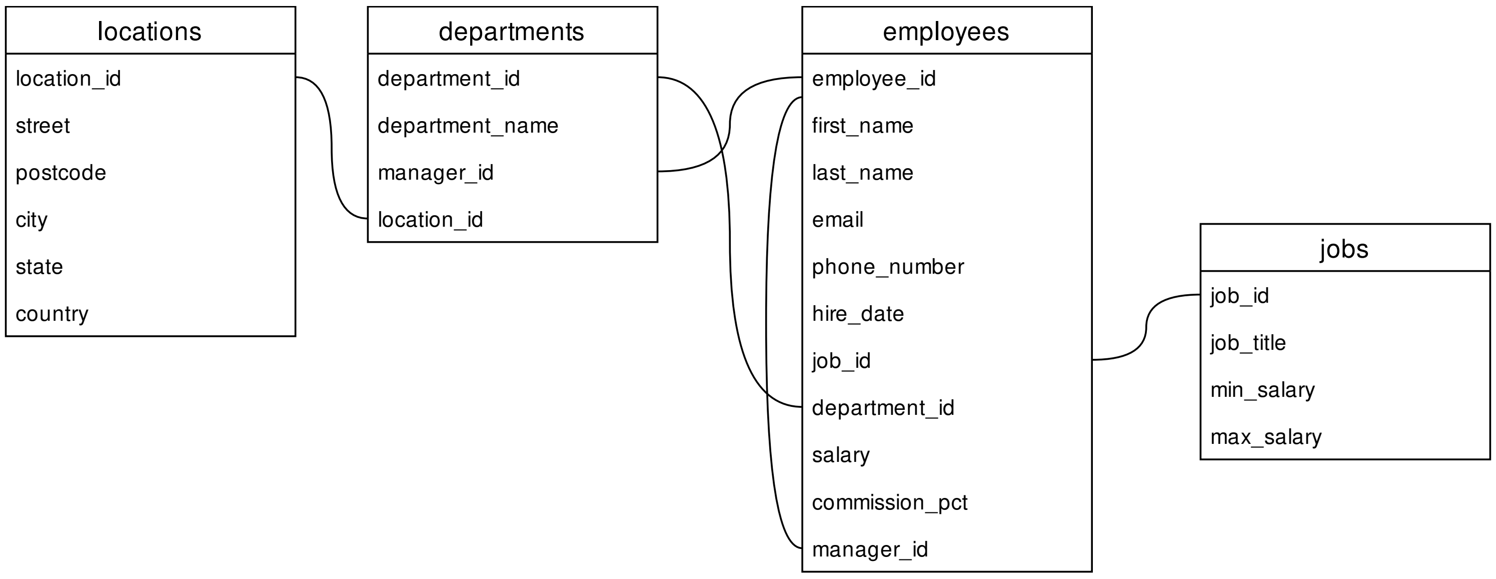
SQL has several parts. Two of the most important are its **Data Definition Language** (DDL), which you can use to define the structure of tables, and its **Data Manipulation Language** (DML), which you can use to insert, modify, retrieve, and delete data in tables.

You will learn a subset of DML - how to retrieve data from your database. You can use these skills not only to query relational databases, but to query non-relational ones as well (as you will see in coming weeks).

Sample database

To help you learn SQL we will be using a sample database: a Human Resources (HR) database. The examples in the slides this week are based on the HR database.

Here is the schema of the HR database (i.e. the structure of its tables):



The database contains information about four kinds of entity: locations, departments, employees, and jobs.

Each **location** has an id number, and address information.

Each **department** has an id number, a name, a manager with a certain id number, and a location with a certain id number.

Each **employee** has an id number, a first name, a last name, an email address, a phone number, a hire date, a job with a certain id number, a department with a certain id number, a salary, a percentage of commission, and a manager with a certain id number.

Each **job** has an id number, a title, and a minimum salary, and a maximum salary.

As we proceed, you'll learn how to use SQL to inspect the actual data that is in these tables.

SQL sandpit

As you learn SQL during these slides you can use the space on the right to write and execute SQL statements to experiment. The sample HR database is loaded for you and ready to query.

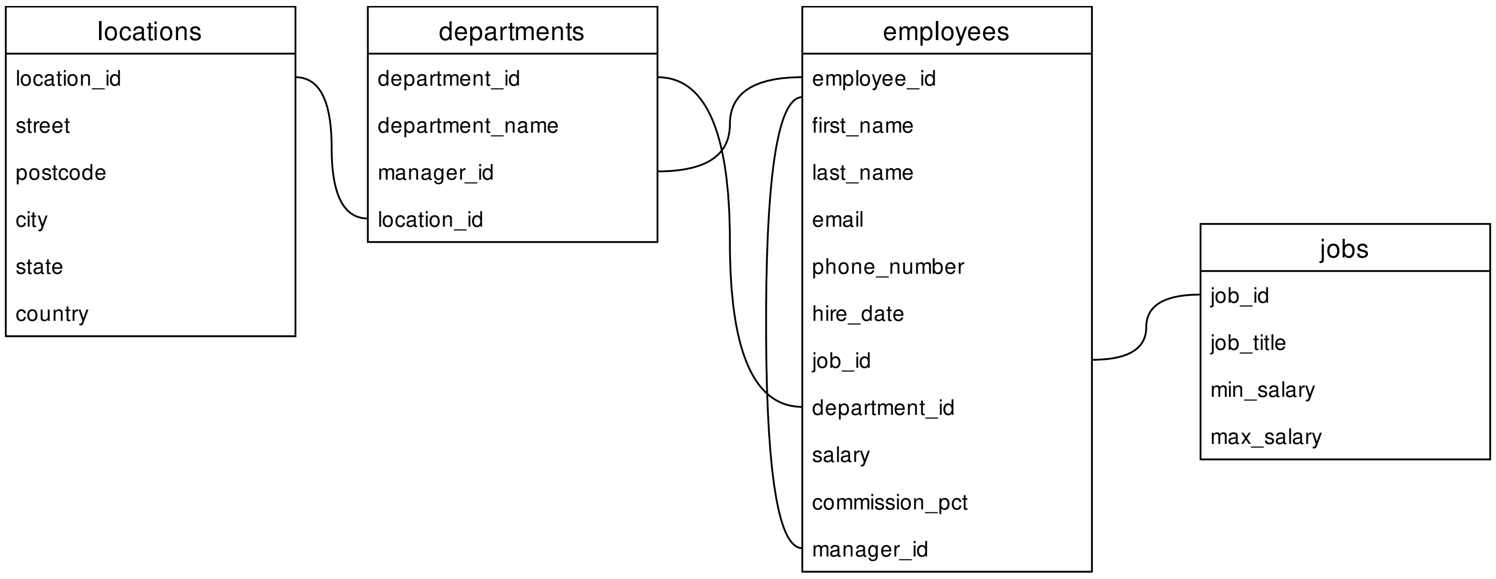
Try the following as an example (you don't need to understand it at this stage - we'll get to that). Copy and paste the following code into the panel on the right and then click "Run":

```
SELECT first_name, last_name FROM employees;
```

The query will be executed and the results of the query will be displayed below it.

You can come back to this slide whenever you like, and experiment with writing your own SQL queries.

To write your SQL statements you need to understand the structure of the HR database. You were given that in the previous slide, but here is the schema again, repeated for your convenience:



Background to SQL

In 1970, E.F. Codd published a paper called "A Relational Model of Data for Large Shared Data Banks", in which he proposed a language for working with data in a relational database.

Based on Codd’s proposal, IBM developed a language called Structured English Query Language (SEQUEL), which later became SQL.

In 1979, Relational Software Inc. (now Oracle) introduced the first commercially available implementation of SQL.

Today, the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO) have both accepted SQL as the standard language for working with relational databases.

There are various dialects of SQL, such as MySQL, PostgresQL, and Oracle SQL. Although they differ slightly, for most of the fundamental parts of the language they are the same. In these slides you will be learning MySQL, but most of what you learn is identical for PostgresQL and Oracle SQL.

A few basics

Syntax

Like other languages, SQL has rules of syntax. There are a few important ones that you need to know:

- SQL has **keywords** that are part of the language and cannot be used as names. For example, "SELECT" and "FROM".
- SQL statements should end with a semi-colon (;). You don't always need to include the semi-colon, but if you get into the habit of doing so then you don't need to worry about whether you do or do not need it.
- Table and column names that have spaces in them must be put in quote marks. For instance, “job history”. It's better to avoid such names in the first place (e.g. rather than naming a column "job history", name it "job_history" or "jobHistory”.
- You can add comments to your SQL code. The rest of a line after two dashes `--` is treated as a comment, and everything between `/*` and `*/` is treated as a comment:

```
-- This is a whole line comment
/* This is a whole line comment */
SELECT salary -- This is an end of line comment
FROM /*This is an inline comment*/ employees;
/*
    This is
    a multiple line
    comment
*/
```

Data types

Every piece of data stored in a relational database has a certain type. There is some variation between RDBMSs, but they all have some version of the following three:

- Numbers: 12, -3.134, 0
- Strings: "Hello", ""
- Dates: "2020-06-24"

NULL

SQL has a special value, NULL, which represents the absence of a value. It is not the same as 0 or "".

Why do we need such a thing? Compare two students, one who sits an exam but gets nothing right, and one who doesn't sit the exam at all. Without NULL we would have to record both of their exam results as 0. But with NULL we can give the first student 0 and the second student NULL, using NULL to represent the lack of a result.

Reading the schema

In these slides we will be working with the HR database. It contains a number of tables. To list the tables you can use the following query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SHOW TABLES;
```

When you click "Run" the query is executed and the results of the query are shown below it. In this case, the query lists all of the tables in the HR database.

To see the structure of the tables you can use these queries:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


DESCRIBE departments;
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
DESCRIBE employees;
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


DESCRIBE jobs;
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


DESCRIBE locations;
```

Each of these queries lists the columns (fields) in the relevant table. It gives various pieces of information about each column. "Field" is the name of the column; "Type" is the type of data that can be stored in the column; "Null" is whether the column can contain null values (don't worry about the other columns in the results).

Selecting columns

To retrieve data from a database you use a SELECT statement. SELECT statements are sometimes called **queries**, because you are querying the database for the answer to a question.

Selecting all columns

To select all columns from a table you can use an asterisk (*) :

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT *
FROM employees;
```

Selecting only some columns

To select only some columns you can name the columns you want, separated by commas. It could be just a single column:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name
FROM employees;
```

Or multiple columns:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees;
```

Try it

Try selecting the minimum and maximum salary columns from the jobs table:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Renaming columns

When you select a column you can give it an **alias**. This will be used as the column heading in the results, and you can also use it to refer to the column in the query itself.

There are four common ways to specify a column alias:

```
SELECT
    column1 AS NewColumn1,
    column2 NewColumn2,
    column3 AS "New Column 3",
    column4 "New Column 4"
FROM
    table_name;
```

The use of the keyword **AS** is optional. If the alias name has spaces then you must surround it with quote marks.

You can use different techniques in the same query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    employee_id AS Employee,
    employee_id AS "Employee Id",
    last_name Surname,
    first_name "First Name"
FROM employees;
```

Try it

Try selecting the department id and name columns from the departments table, calling the first column "Id" and the second column "Department name":

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Removing duplicate rows

By default a SELECT statement will return all matching rows, even if some are duplicates of others. For example, the following statement returns all job_ids, even though some are duplicates of others:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT job_id
FROM employees;
```

If you only want to return unique rows, and eliminate any duplicates, you can use the DISTINCT keyword. The following query returns a list of *distinct* job_ids:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT DISTINCT job_id
FROM employees;
```

If you use the DISTINCT keyword when selecting multiple columns, the results will include the unique *combinations* of those columns:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
```

```
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT DISTINCT department_id, manager_id
FROM employees;
```

Try it

Try selecting the distinct combinations of minimum and maximum salaries from the jobs table:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Sorting rows

You can sort the rows returned by a query using an ORDER BY clause in your query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT employee_id, last_name, hire_date
FROM employees
ORDER BY last_name;
```

The ORDER BY clause should always be the last clause in your SELECT statement.

Sorting by multiple columns

You can sort by multiple columns. The following query sorts rows first by last_name and then by first_name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT employee_id, last_name, first_name
FROM employees
ORDER BY last_name, first_name;
```

Sort direction

By default, sorting is done in ascending order. If you want to sort in descending order you can add

DESC after the column name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, manager_id, hire_date
FROM employees
ORDER BY manager_id, hire_date DESC;
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
ORDER BY last_name DESC, first_name;
```

You can add ASC also, to sort in ascending order, but this is the default so you don't need to.

Sorting by unselected columns

You can sort by a column that is not on the SELECT list. In the example below, the manager_id column is not listed in the SELECT clause:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, hire_date
```

```
FROM employees
ORDER BY manager_id, hire_date DESC;
```

NULL values

When you sort in ascending order, NULL values appear at the top:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, manager_id, commission_pct
FROM employees
WHERE manager_id = 100
ORDER BY commission_pct;
```

If you want NULL values to be at the bottom, you can first sort rows by IS NULL:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, manager_id, commission_pct
FROM employees
WHERE manager_id = 100
ORDER BY commission_pct IS NULL, commission_pct;
```

Try it

Try selecting city and postcode from the locations table, sorted first by state in descending order, then by city, then by postcode:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Manipulating columns

You can manipulate columns by combining them or performing calculations.

Concatenating columns

You can combine columns by concatenating them, using the `CONCAT()` function (some versions of SQL allow you to use `+` or `||`):

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT CONCAT(first_name, last_name)
FROM employees;
```

You can concatenate as many columns as you like, and you can add literal values too:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT CONCAT("Full name: ", first_name, " ", last_name)
FROM employees;
```

When using numeric literals you should not use quotes. You can use single quotes inside double quotes, and double quotes inside single quotes:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
```

```
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT CONCAT(first_name, "'s last name is ", '"', last_name, '"')
FROM employees;
```

Performing calculations

For the basic arithmetic operations of addition, subtraction, multiplication and division, use `+`, `-`, `*` and `/`:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    last_name,
    salary,
    salary * 0.10 AS "Increase",
    salary + (salary * 0.10) AS "New Salary"
FROM employees;
```

Be careful with order of operations. For example, if you have the expression `salary + 12 * 100`, this will be interpreted as `salary + (12 * 100)` rather than as `(salary + 12) * 100`, because multiplication takes **precedence** over addition. If you intend the latter calculation then you need to use parentheses in your expression to make that clear: use `(salary + 12) * 100` instead of `salary + 12 * 100`.

It's good practice to always use parentheses, to help avoid unintended mistakes, and to make your intentions clear to anyone who might read your SQL code (which could be you, in the future).

Here are the orders of precedence of the four basic operations:

First **multiplication**, then **division**, then **addition**, then **subtraction**.

Working with NULLs

When working with NULL values there are some things to be aware of.

Any number multiplied by NULL is NULL

```
SELECT 10 * NULL, NULL * (34 + 123);
```

You can use IFNULL to replace NULL values with 0:

```
SELECT IFNULL(10, 0), IFNULL(NULL, 0);
```

Now you can get zero instead of NULL:

```
SELECT 10 * IFNULL(NULL, 0);
```

Try it

Try selecting employees, showing their "reverse name" (e.g. "Pitt, Brad") and total salary once commission is included, sorted by reverse name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

String functions

There are various functions that you can use with strings.

In many of the examples below a SELECT statement is executed without any reference to a table. This is perfectly fine, as long as you are just working with non-database values. It's a bit like using SQL as a calculator.

To get the length of a string:

```
SELECT LENGTH('Hello');
```

To change the case of a string:

```
SELECT UPPER('Hello'), LOWER('Hello');
```

To find the first occurrence of a substring in a string:

```
-- Returns the position (index) of the first occurrence of 'el'
-- Indexes start at 1
SELECT INSTR('Hello', 'el');
```

To find the substring at a given position in a string:

```
-- Returns the substring that starts at index 2 and has length 3
SELECT SUBSTR('Hello', 2, 3);
```

To replace part of a string:

```
-- Replaces all occurrences of 'l' by 'x'
SELECT REPLACE('Hello', 'l', 'x');
```

To trim white space from a string:

```
-- Removes white space from left, right, and both, respectively
SELECT LTRIM(' Hello '), RTRIM(' Hello '), TRIM(' Hello ');
```

To pad a string to a fixed length:

```
-- Left/right pads to length 10 using 'x',
SELECT LPAD('Hello', 10, 'x'), RPAD('Hello', 10, 'x');
```

More functions

For a full list of MySQL string functions see the following:

w3schools list of MySQL functions

MySQL documentation list of string functions

Try it

Try selecting the first name, last name, and initials of employees, ordered by last name then first name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Numeric functions

There are various functions that you can use to work with numbers.

To round a number to a given number of decimal places:

```
-- Optional number of decimal places
SELECT ROUND(3.247), ROUND(3.247, 1), ROUND(3.247, 2);
```

To round a number up or down to the nearest integer:

```
SELECT CEILING(3.247), FLOOR(3.247);
```

To truncate a number to a given number of decimal places:

```
-- The second argument is required
SELECT TRUNCATE(3.247, 0), TRUNCATE(3.247, 1), TRUNCATE(3.247, 2);
```

To find the absolute value of a number:

```
SELECT ABS(0), ABS(3.247), ABS(-3.247);
```

To raise one number to the power of another:

```
-- Raise the first argument to the power of the second argument:
-- Notice that negative and fractional powers work
SELECT POWER(2, 3), POWER(4, -1), POWER(16, 0.5);
```

To find the remainder when one number is divided by another:

```
-- Find the remainder when 10 is divided by 3:
-- Note there are a few different ways
SELECT MOD(10, 3), 10 MOD 3, 10 % 3;
```

More functions

For a full list of MySQL numeric functions see the following:

w3schools list of MySQL functions

MySQL documentation list of numeric functions

Try it

Try selecting a list of jobs showing each job's title, its salary range (i.e. the difference between the maximum and minimum salaries), and what percentage of the minimum salary that range is (to the nearest percent):

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Date functions

Manipulating dates is not an easy task. Some global organisations have systems that are used across different countries and timezones where the date format may differ in each country. For instance, in Australia, the default format for dates is dd-mm-yyyy (e.g. 18-01-2011) whereas, in the United States, the format is mm-dd-yyyy (e.g. 01-18-2011). RDBMSs can handle the different date formats used in different countries.

The default format of a date in MySQL is YYYY-MM-DD. Different SQL languages handle dates differently, so you have to check the date format, function and how the date is handled in that language. In Oracle it is DD-MON-RR, where "MON" stands for a three letter month name and "RR" stands for to a two digit year number.

To get the current date:

```
SELECT CURDATE();
```

To get the number of days between two dates:

```
-- First date minus the second date, expressed in days
-- You would normally have the latest date first
SELECT
    DATEDIFF('2020-06-10', '2020-06-10') AS 'A',
    DATEDIFF('2020-06-10', '2020-06-12') AS 'B',
    DATEDIFF('2020-06-10', '2020-06-08') AS 'C';
```

To get the various parts of a date:

```
SELECT
    DAYNAME('1968-06-24') AS 'DayName',
    DAY('1968-06-24') AS 'Day',
    MONTH('1968-06-24') AS 'Month',
    MONTHNAME('1968-06-24') AS 'MonthName',
    YEAR('1968-06-24') AS 'Year';
```

To format a date in a certain way:

```
SELECT DATE_FORMAT('1968-06-24', '%W, %D %M, %Y');
```

What do all those symbols mean? Here is a good reference.

More functions

For a full list of MySQL date functions see the following:

w3schools list of MySQL functions

MySQL documentation list of date (and) functions

Try it

Try selecting a list of employees showing each employee's first name, last name, and hire date formatted like "June 24, 1968 (Monday)".

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Other functions

The CASE function

The CASE function provides a convenient way to return a value that depends upon which of a number of conditions obtains:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    last_name,
    Salary,
    CASE
        WHEN (salary >= 10000) THEN 'Level 5'
        WHEN (salary >= 8000) THEN 'Level 4'
        WHEN (salary >= 5000) THEN 'Level 3'
        WHEN (salary >= 2500) THEN 'Level 2'
        ELSE 'Level 1'
    END AS Level
FROM employees
ORDER BY salary DESC;
```

Try it

Try selecting a list of locations showing each location's country, and the word "Local" if the country is Australia, "North America" if the country is USA or Canada, and "Other" for other countries, ordered by country.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Filtering rows

You can select only certain rows by adding a WHERE clause to your query.

The following query selects employees whose salary is greater than $10,000.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, salary, hire_date
FROM employees
WHERE salary > 10000;
```

The following query selects employees whose last name is "King"..

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, salary, hire_date
FROM employees
WHERE last_name = 'King';
```

The following query selects employees who were hired on or before 8th March 2008.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
SELECT employee_id, last_name, first_name, salary, hire_date
FROM employees
WHERE hire_date <= '2008-03-08';
```

When filtering string or date columns you should use quotes around the filtering value, but not for numeric columns. For dates, make sure you are aware of the format in which dates are stored. In MySQL the default is YYYY-MM-DD (e.g. 2011-01-16). In Oracle it is DD-MON-YY (e.g. 16-JAN-11).

Filtering strings is NOT case sensitive in MySQL - filtering by 'King' produces the same results as filtering by 'KING'. Be aware that in some versions of SQL the filtering is case sensitive.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name = 'King';
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name = 'KING';
```

Try it

Try selecting all locations in the USA, showing each location's state and city, ordered by state then city:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Comparing values

Comparison operators can be used in a WHERE clause to compare columns, literal values and/or expressions. You've already seen examples using the operators equal (=) and greater than (>). Now let's look at the other comparison operators you can use.

The following query selects employees who were hired on or after 8th March 2008:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, hire_date
FROM employees
WHERE hire_date >= '2008-03-08';
```

The following query selects departments whose location id is not 1700:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
SELECT department_id, department_name, location_id
FROM departments
WHERE location_id <> 1700;
```

The following query selects employees whose department id is NULL:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, department_id
FROM employees
WHERE department_id IS NULL;
```

Adding NOT

You can filter for records that do *not* satisfy a certain condition by adding the NOT operator to the condition. For example, to find employees whose department id is *not* 40 or 60 you could use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, department_id
FROM employees
WHERE department_id NOT IN (40, 60);
```

Combining with AND and OR

You can filter for records that satisfy **both** of two conditions by using the AND operator.

To find employees whose salary is greater than $10,000 and less than $11,000 you could use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, salary
FROM employees
WHERE salary > 10000 AND salary < 11000;
```

To find employees whose salary is at least $10,000 and no more than $11,000 you could use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, salary
FROM employees
WHERE salary >= 10000 AND salary <= 11000;
```

To find employees whose id starts with 1, has any number in the middle, and ends with 9, and who was hired in the year 2008, you could use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, hire_date
FROM employees
WHERE employee_id LIKE '1_9' AND hire_date LIKE '2008-%';
```

You can filter for records that satisfy **either** of two conditions (or both) by using the OR operator.

To find employees whose last name starts with 'A' or 'B' you could use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name LIKE 'A%' OR last_name LIKE 'B%';
```

Beware of ambiguity

When you use an expression that contains two or more operators then it is a good idea to use parentheses to make it clear how you intend them to be grouped. For example:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE (last_name LIKE 'A%' AND salary > 10000) OR last_name LIKE 'B%';
```

Comparing strings

When a comparison is done between two strings, the values are compared using ASCII codes. ASCII stands for American Standard Code for Information Interchange. ASCII codes represent a number that computer or software can translate. For example, the ASCII for character 1 is 49 (decimal).

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT location_id, postcode, country
FROM locations
WHERE postcode = 'YSW 9T2';
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT location_id, postcode, country
FROM locations
WHERE postcode > '9';
```

As you can see, 'M5V 2L7', 'YSW 9T2', and 'OX9 9ZB' are among the selected records. This is because the ASCII decimal value for the number nine is 57 and, as an example, the ASCII decimal value for the letter 'Y' is 89. Since 89 is greater than 57, these values are among the selected records.

BETWEEN ... AND ... operator

To select values in a certain range you can use a combination of operators as follows:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name >= 'King' AND last_name <= 'Lee';
```

Alternatively, you can use the BETWEEN ... AND ... operator:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name BETWEEN 'King' AND 'Lee';
```

You must specify the lower limit first - the following query returns no results:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name BETWEEN 'Lee' AND 'King';
```

Try it

Try selecting all employees whose last name starts with 'A' or 'B' and whose salary is between $5000 and $10,000 (inclusive), showing each employee's last name and salary, ordered by last name ascending then salary descending:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Pattern matching

You can use the LIKE operator and its associated wildcard characters to filter to values that match a certain pattern.

There are two wildcard characters that you can use:

- Percent sign (%): matches zero or more characters, e.g. "bl%" matches "bl", "black", "blue", and "blob"
- Underscore sign (_): matches exactly one character, e.g. “h_t” matches "hot", "hat", and "hit".

For example, the following query returns all employees whose surname starts with 'S'.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name LIKE 'S%';
```

You can combine uses of LIKE. The following query returns employees whose employee id starts with 1, has any number in the middle, and ends with a 9, and who started working in the year 2008:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name, hire_date
FROM employees
WHERE employee_id LIKE '1_9' AND hire_date LIKE '2008-%';
```

If you'd like the employees whose surname does *not* start with 'S', you could add the NOT keyword:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT employee_id, last_name, first_name
FROM employees
WHERE last_name NOT LIKE 'S%';
```

Other wildcards

Some RDBMSs have other wildcards that you can use:

- Character(s) within the bracket sign ([]): match character(s) with character(s) within the bracket, e.g. “h[oa]t” finds hot and hat, but not hit
- Not Character(s) within the bracket sign (^): match any character not within the brackets, e.g. h[^oa]t finds hit, but not hot and hat
- A range of characters (-): match a range of characters, e.g. c[a-u]t finds cat and cut

Try it

Try selecting all locations whose city starts with "South" or has "City" in it (or both), showing each location's city, ordered by city:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Aggregate functions

There are special functions that you can use to get totals, called **aggregate functions**.

COUNT

To find the total number of rows in a table you can use COUNT(*):

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT COUNT(*)
FROM employees;
```

To find the total number of department_ids in the employees table you can use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT COUNT(department_id)
FROM employees;
```

Notice that this only returns 106. Why not 107, as for the first query? Because one of the department_ids is NULL, and that doesn't get counted. You can verify there is one such row using the following query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT *
FROM employees
WHERE department_id IS NULL;
```

If you just want the total number of *distinct* department_ids in the employees table you can add the keyword DISTINCT:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT COUNT(DISTINCT department_id)
FROM employees;
```

MIN, MAX, and AVG

To find the minimum, maximum, and average salaries in the employees table you can use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT MIN(salary), MAX(salary), AVG(salary)
FROM employees;
```

SUM

To find the sum of salaries in the employees table you can use:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT SUM(salary)
FROM employees;
```

You can verify that AVG() gets the right results by doing the calculation from first principles:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT AVG(salary), SUM(salary)/COUNT(salary)
FROM employees;
```

GROUP_CONCAT

Rather than counting or summing the values you can combine them into a string, using the GROUP_CONCAT function:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT GROUP_CONCAT(city)
FROM locations
```

By default, a comma is used as the separator. You can specify an alternative by using the SEPARATOR keyword:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT GROUP_CONCAT(city SEPARATOR '; ')
FROM locations
```

You can also specify how you would like the items to be ordered, by adding ORDER BY (and DESC, if you want descending order):

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT GROUP_CONCAT(city ORDER BY city DESC SEPARATOR '; ')
FROM locations
```

Does it matter what order you use ORDER BY and SEPARATOR? Try for yourself:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT GROUP_CONCAT(city SEPARATOR '; ' ORDER BY city DESC)
FROM locations
```

Working with NULLs

Be careful when using aggregate functions on columns that contain NULL values. The `AVG()` function, for example, ignores NULL values when calculating averages.

Here's an example where this matters. Each of the two queries below calculates the average commission paid to employees, but they treat NULL values of commission_pct differently. The first ignores them, whereas the second counts them as 0. Thus, the first query returns a higher average than the second. In a bit more detail: The first query leaves NULL values of commission_pct as NULL. When salary is multiplied by NULL the result is NULL, and these NULLs are ignored by the `AVG()` function. The second query converts NULL values of commission_pct to 0. When salary is multiplied by 0 the result is 0, and these 0s are included in the `AVG()` function.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



-- Leave NULL values of commission_pct as NULL
SELECT
    COUNT(*) AS total_rows,
    COUNT(salary * commission_pct) AS included_rows,
    AVG(salary * commission_pct) AS average
FROM employees
```

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



-- Change NULL values of commission_pct into 0
SELECT
    COUNT(*) AS total_rows,
    COUNT(salary * IFNULL(commission_pct, 0)) AS included_rows,
    AVG(salary * IFNULL(commission_pct, 0)) AS average
FROM employees
```

Try it

Try selecting the number of distinct cities in the locations table:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Grouping rows

When using aggregate functions to get totals you typically want the totals for *groups* of rows, not for all the rows in the table. You can achieve this by adding a GROUP BY clause.

The following query returns the total number of employees:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT COUNT(*)
FROM employees;
```

If we want to know the number of employees in each department, we can add a GROUP BY clause that groups by department_id:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT COUNT(*)
FROM employees
GROUP BY department_id;
```

It's more illuminating if we also show the department_id column, and sort by total number in descending order:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
```

```
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT department_id, COUNT(*) AS total
FROM employees
GROUP BY department_id
ORDER BY total DESC;
```

If you only want to count the employees whose salary is greater than $2000 you can add a WHERE clause:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT department_id, COUNT(*) AS total
FROM employees
WHERE salary > 3000
GROUP BY department_id
ORDER BY total DESC;
```

The query applies this WHERE clause to get the relevant rows, and then groups and totals them.

HAVING

What if you want to filter the results *after* the grouping and totalling is done? For example, what if you only want the number of employees in the departments that have at least 10 employees?

In this case you can use a HAVING clause:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
SELECT department_id, COUNT(*) AS total
FROM employees
GROUP BY department_id
HAVING total >= 10
ORDER BY total DESC;
```

A HAVING clause is like a WHERE clause, but it filters results *after* doing the grouping and totalling.

You can have both a WHERE clause and a HAVING clause. Both filter the results, but the WHERE clause is applied before grouping and totalling, while the HAVING clause is applied after.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT department_id, COUNT(*) AS total
FROM employees
WHERE salary > 3000
GROUP BY department_id
HAVING total >= 10
ORDER BY total DESC;
```

Try it

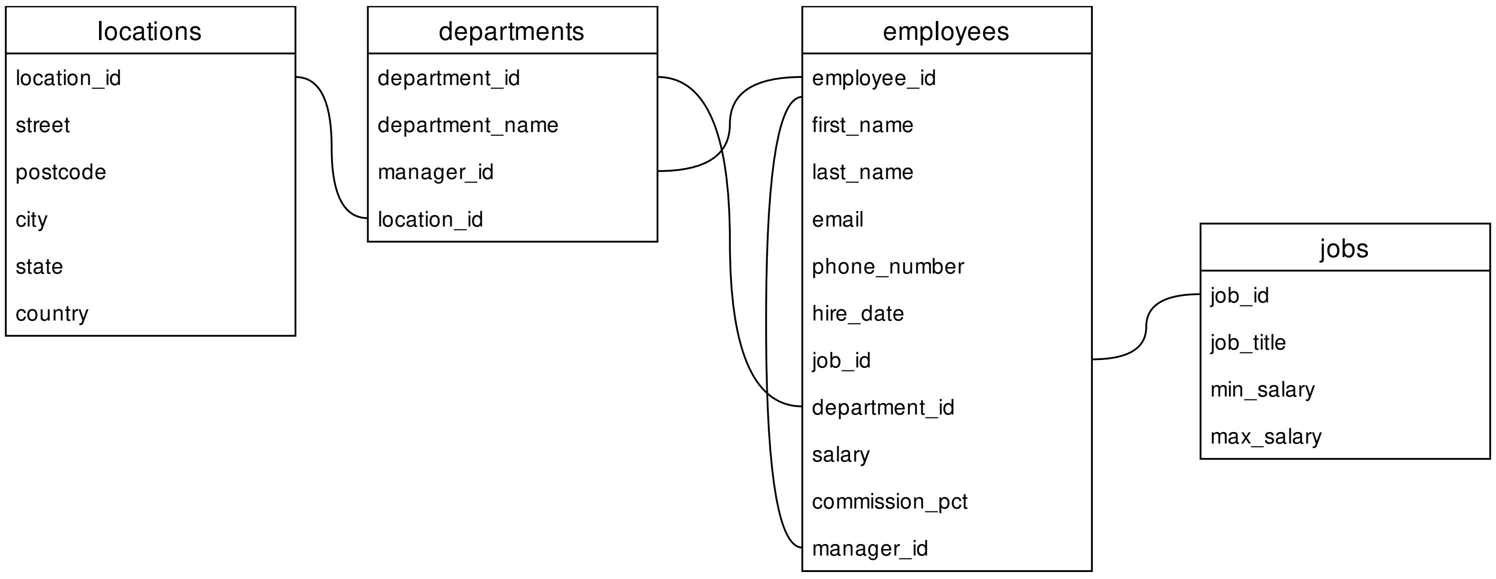
Try selecting a list of countries from the locations table, showing the country, the number of cities in that country, and a list of those cities (separated by ", " and in alphabetical order), ordered by country:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Querying multiple tables

So far we have been querying the data in just a single table. Often, however, the data that you want is spread across two or more tables.

Suppose, for example, that you want a list of departments, showing their id, name, and city. The id and name of each department are kept in the departments table, but the city is not. Rather, a location id is stored there, and the city is stored in a corresponding row in the locations table. (This happened in the process of normalising the database.) You can see this in the database diagram:



Since the city is not stored in the departments table, the following query won't work (try it):

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT department_id, department_name, city
FROM departments;
```

INNER JOIN ... USING (...)

Instead, you need to **join** the department and locations tables, on the field that connects them - the location_id field. Then the city column is available for selecting:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT department_id, department_name, city
FROM departments INNER JOIN locations USING (location_id);
```

The word "INNER" is used to distinguish the join from an OUTER join - we'll see what the difference is in the next slide.

INNER JOIN ... ON (...)

Rather than using JOIN ... USING (...) to join tables you can use JOIN ... ON (...) to specify column to join:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    department_id,
    department_name, city
FROM
    departments
    INNER JOIN locations ON (departments.location_id = locations.location_id);
```

This is more verbose. So why would you use it? Sometimes the linking fields have a different name in the two tables, and in that case you have to use JOIN ON -- JOIN USING won't work.

Resolving ambiguities

If you add location_id to the SELECT clause in the previous query it will give rise to an error:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    department_id,
    department_name,
    location_id,
    city
FROM
    departments
    INNER JOIN locations ON (departments.location_id = locations.location_id);
```

The problem is that there is a field called "location_id" in both the departments table and the locations table, and when we refer to "location_id" in the query the query interpreter doesn't know which one we mean - there is an **ambiguity**.

To resolve this ambiguity we just make it explicit which table we intend, by preceding the field name with the table name, just as we have done in the ON clause: "departments.location_id".

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    department_id,
    department_name,
    departments.location_id,
    city
FROM
    departments
    INNER JOIN locations ON (departments.location_id = locations.location_id);
```

It can be a good idea to do this for *all* fields, to make sure there is no risk of ambiguity:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    departments.department_id,
    departments.department_name,
    departments.location_id,
    locations.city
FROM
    departments
    INNER JOIN locations ON (departments.location_id = locations.location_id);
```

Three or more tables

Sometimes the data you want is spread across three or more tables. You can join them all together in the same way:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    first_name,
    last_name,
    department_name,
    city
FROM
    employees
    INNER JOIN departments ON (employees.department_id = departments.department_id)
    INNER JOIN locations ON departments.location_id = locations.location_id
WHERE
```

```
    employees.department_id = 90;
```

Self-Joins

It's perfectly fine to join a table to itself, and in fact sometimes you need to do so.

For example, suppose you want a list of employees in department 90, showing their name, and also the name of their manager. The id of the manager is kept in the manager_id field. To get the name of the manager you will have to join the employees table with itself, connecting the manager_id field in one with the employee_id field in the other:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    employees.first_name,
    employees.last_name,
    managers.first_name,
    managers.last_name
FROM
    employees
    INNER JOIN employees as managers ON (employees.manager_id = managers.employee_id)
WHERE
    employees.department_id = 90;
```

Note that in this case you must use an alias for at least one of the employees tables, otherwise you won't be able to refer to the tables unambiguously.

To help distinguish the names in the output, it's a good idea to introduce an alias for each column selected:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
SELECT
    employees.first_name AS emp_first_name,
    employees.last_name AS emp_last_name,
    managers.first_name AS man_first_name,
    managers.last_name AS man_last_name
FROM
    employees
    INNER JOIN employees as managers ON (employees.manager_id = managers.employee_id)
WHERE
    employees.department_id = 90;
```

Try it

Try selecting a list of employees, showing each employee's first name, last name, job title, department name, and department city. Order by city, department name, last name, and first name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Outer joins

So far we have been using INNER joins. You use an INNER join when you only want to include rows that have a matching record in the tables on both sides of the join.

Sometimes you want to include rows that do not have a match in the other table. In this case, you use an OUTER join. You can think of the outer join as an *optional* link to another table.

There are two types of OUTER join: LEFT and RIGHT.

LEFT OUTER JOIN

With a LEFT OUTER JOIN, records in the table on the left side of the join will be included in the results, even if they have no matching record in the table on the right side.

The employee Steven King does not have a manager (he is the president of the company). If you use an INNER JOIN, he will not be included in the results:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT
    employees.first_name,
    employees.last_name,
    managers.first_name,
    managers.last_name
FROM
    employees
    INNER JOIN employees as managers ON (employees.manager_id = managers.employee_id)
WHERE
    employees.last_name LIKE 'K%';
```

If you use a LEFT OUTER join, he will be included. Notice that the information about his manager is all NULLs.

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    employees.first_name,
    employees.last_name,
    managers.first_name,
    managers.last_name
FROM
    employees
    LEFT OUTER JOIN employees as managers ON (employees.manager_id = managers.employee_id)
WHERE
    employees.last_name LIKE 'K%';
```

RIGHT OUTER JOIN

With a RIGHT OUTER join, records in the table on the right side of the join will be included in the results, even if they have no matching record in the table on the left side.

Suppose you want a list of departments and their locations, but you want to include locations even if there is no department there. You could use a RIGHT OUTER join as follows:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091



SELECT
    department_name,
    city
FROM
    departments
    RIGHT OUTER JOIN locations USING (location_id)
ORDER BY
    department_name, city;
```

Try it

Try selecting a list of departments that have no employees, showing the name of the department, sorted by name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Subqueries

Sometimes you will want to use a query within your query. Such a query is known as a **subquery**. There are a couple of different ways you might use a subquery - as a value, or as a table.

Subquery as value

Suppose you want a list of the employees who have the largest salary. If you know what that salary is, let's say $10,000 then you could use the following query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name, first_name, salary
FROM employees
WHERE salary = 10000
ORDER BY last_name, first_name;
```

But what if you don't know what the largest salary is? Then in place of "10000" you could use a query that finds the largest salary for you:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name, first_name, salary
FROM employees
WHERE salary = (SELECT MAX(salary) FROM employees)
ORDER BY last_name, first_name;
```

Here we have a query within a query, which is called a **subquery**. (Can you think of a way to do this last query without using a subquery?)

Subquery as table

Suppose, instead, that you want a list of employees who have one of the three largest salaries (rather than just the largest salary). Again, if you knew what those three largest salaries were, let's say $10,000, $9,000 and $8,000, then you could use the following query:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name, first_name, salary
FROM employees
WHERE salary IN (10000, 9000, 8000)
ORDER BY last_name, first_name;
```

But what if you don't know what the three largest salaries are? You could try using a query that finds them for you, as we did above, but unfortunately it won't work in this case:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name, first_name, salary
FROM employees
WHERE salary IN (
    SELECT DISTINCT salary FROM employees ORDER BY salary DESC LIMIT 3
)
ORDER BY last_name, first_name;
```

What you can do, instead, is use this subquery as if it were a table, and join it to the employees table:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091


SELECT last_name, first_name, salary
FROM
    employees
    INNER JOIN (
        SELECT DISTINCT salary FROM employees ORDER BY salary DESC LIMIT 3
    ) AS TopThreeSalaries USING (salary)
ORDER BY last_name, first_name;
```

By using an INNER JOIN between the two we can restrict our list of employees to just those whose salary is one of the top three.

Note that when you use a subquery as a table you must give it an alias, even if you never use that alias. In this case we have called it "TopThreeSalaries".

Try it

Try selecting a list of employees that are in the department which has the greatest number of employees. Show department name, employee last name, employee first name, ordered by department name, last name, first name:

```
CREATE TABLE departments (department_id int,department_name tinytext,manager_id int,location_id int
INSERT INTO departments VALUES (10,'Administration',200,1700),(20,'Marketing',201,1800),(30,'Purcha
CREATE TABLE employees (employee_id int,first_name tinytext,last_name tinytext,email tinytext,phone
INSERT INTO employees VALUES (100,'Steven','King','SKING','515.123.4567','2003-06-17',1,90,24000,NU
CREATE TABLE jobs (job_id int,job_title tinytext,min_salary int,max_salary int);
INSERT INTO jobs VALUES (1,'President',20080,40000),(2,'Administration Vice President',15000,30000)
CREATE TABLE locations (location_id int,street tinytext,postcode tinytext,city tinytext,state tinyt
INSERT INTO locations VALUES (1000,'1297 Via Cola di Rie','00989','Roma',NULL,'Italy'),(1100,'93091
```

Further resources

For a gentle guide to SQL the tutorials at w3schools are very good:

w3schools tutorial on SQL

If you'd like to see a more theoretical discussion you could look at **Section 4.3 of Elmasri and Navathe**, "Fundamentals of Database Systems" (this book is on the course's reading list, which you can find in Moodle).