Introduction

What is SQL?

SQL (pronounced ess-cue-ell) is the standard database language, supported to some extent by every database product on the market today. It has an internationally agreed syntax defined in a document which is well over 600 pages long. Most companies implement a subset of the standard and introduce minor additions to enable it to work with their specific database product. SQL is a fourth generation language (4GL), for reasons that will be discussed later.

History of SQL

Until the mid 1980's, almost all mainframe databases were either hierarchical or network databases, which lacked a rigorous mathematical foundation and were often implemented in a messy or inefficient style. In 1970, E.F. Codd published a paper which provided a relational model for databases based on set theory. It took ten years for IBM to produce System R based on this relational model, which included a query language called *sequel*. During the period 1980 to 1983 IBM announced versions for various Operating Systems and machines. Such was the power of IBM at the time that many other companies announced a version of SQL for their machines, either as a replacement or as an alternative for their proprietary language.

The Standards Bodies

To prevent there from being multitudes of versions with differing dialects, ANSI and the ISO standards committee have produced documents detailing various standards and extensions, SQL1, SQL2 (SQL92), SQL 2003 and SQL 2006. SQL1 was produced in its initial form in 1987. SQL 92 is the most common version and is detailed in

- ISO/IEC 9075:1992, "Information Technology --- Database Languages --- SQL"
- ANSI X3.135-1992, "Database Language SQL"

The current standard is SQL 2006, which can still be regarded as a draft. Few databases implement all the current features.

Further information about the current state of the standard can be found on the web – unfortunately formal versions from SQL92 onwards are copyright so it can be difficult to obtain copies. Descriptions of SQL86 seem to be more freely available.

The structure of a database

The Relational Model

To use SQL, the programmer must be familiar with the relational model, which is at the heart of all RDBMS's (Relational Database Management Systems). In simple terms, data is manipulated in *tables* (sometimes called *Relations*), at a conceptional level several layers above how the data is actually stored. SQL users don't have to know anything about files, pointers, links or blocks.

The Table/Relation concept

`EMP table`

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
405	MARCH	ADMIN	938	13-Jun-97	18000		2
936	CASSY	ADMIN	734	23-Jul-02	19500		3
912	HAYES	ADMIN	824	04-Jun-01	21000		2
557	BELL	SALES	734	26-Mar-00	22500	500	3
690	AHMAD	SALES	734	05-Dec-97	22500	1400	3

Each table must have a unique name (EMP in this case). Each row (record) is a series of interconnected data items, an employee in this case. This snapshot of EMP table shows 5 rows/5 employees. A cell must hold one Atomic value (i.e. a value that wouldn't normally be divided into any smaller parts). Values can be Text (i.e. letters or Alphanumeric characters), Numbers (so that associated mathematical operations can be performed) or other types such as Dates, Times or Currency.

The table rows must be 'interrelated' in some way. The EMP table contains a set of employees in a company. Each column must have a unique column name (to that table), which indicates the kind of data items shown in the column 'below'.

Every table is supposed to mirror a mathematical set and as such there is no significance in the row or column ordering. Theoretically there are also no duplicate rows allowed (actual databases may allow duplicates). A *table* is sometimes called a *relation* in mathematics.

Operations on a table are based on the mathematical principles of *selection*, *projection*, *join* and *product* (these terms come from an area of mathematics called Relational Algebra). In an actual database these operations are performed through a language such as SQL. SQL is a non procedural language which has an English like structure and allows the user to specify what is wanted rather than how the computer should perform the task (the major difference between a third and fourth generation language).

Microsoft Access

Starting Access

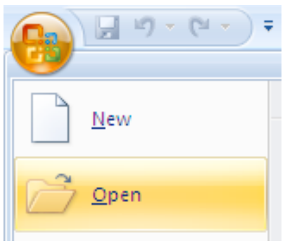
In Windows the Access database package can be started by finding the Access menu item (which is typically under the Microsoft Office menu) or by clicking on the Access icon if one is visible on the screen.

Activity 1: The first exercises in these notes use the database ***session1 2007.mdb***

This is available on the web at

```
http://www.barryavery.com/blog/teaching/database/
```

After saving this file in your network area, click on the Office Button -> Open to start a database session and select *session1 2007.accdb* from where the file was saved.



Alternatively you can double click on the *session1 2007.accdb* file, which should start Access and attempt to load the database.

The file format

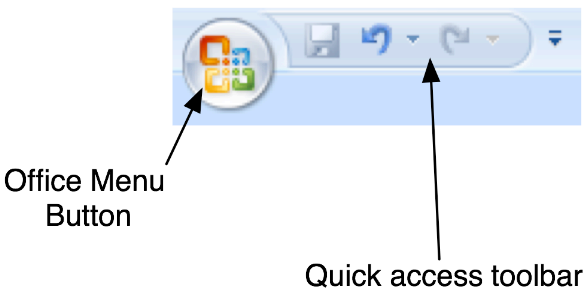
Microsoft Access saves each database in a single file with the extension *.accdb*. Note that this is unusual - most other databases save information in several files with differing file extensions. Access file formats have changed significantly between versions. Versions in use range from Access 97 through to Access 2007.

Files created in a previous version of Access can (normally) be opened as read-only in the newer version, or alternatively transformed through some process to the latest file format. Once the changes are saved it may no longer be possible to open the file in the previous version.

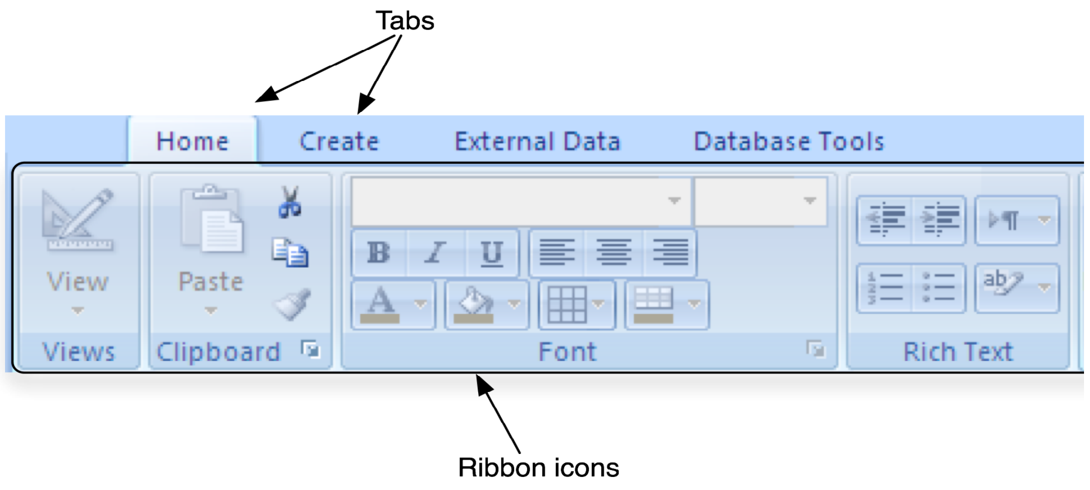
Office 2007

The latest version of Office has a significantly different interface to previous versions. It introduces an Office button, the ribbon and context generated tabs of icons.

The Office Menu Button replaces many of the functions that were originally available from the File menu under older versions. The Quick Access Toolbar has icons for Save, Undo and Redo



The ribbon (common to all applications in the Office 2007 suite), has tabs and icons for the different functions and features. There are four default tabs, although Access will add/remove context sentive tabs as it determines is appropriate.

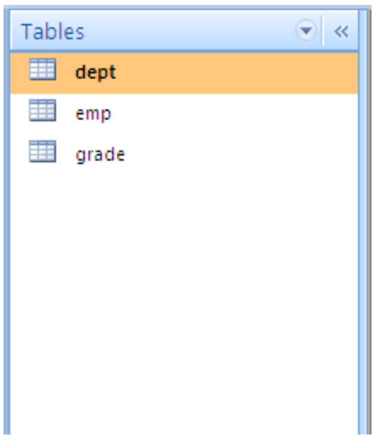


- *Home* has copy/paste, formatting tools for fonts/colours, sorting, searching and filtering
- *Create* has commands for inserting different database objects including tables, queries, forms and reports
- *External Data* has commands for importing data into Access and exporting it to other programs.
- *Database Tools* has professional tools used to analyze a database, link tables and scale up to SQL Server as well as facilities for macro and Visual Basic programming.

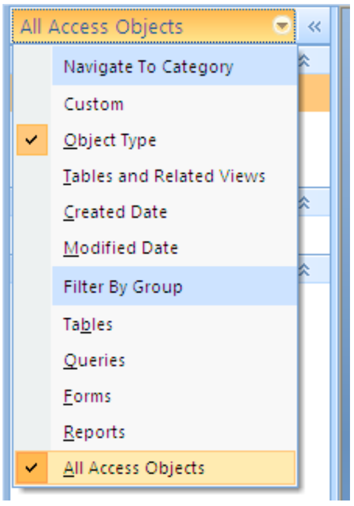
The Navigation pane

Access 2007 uses a navigation pane to manage the different database objects you create and use.

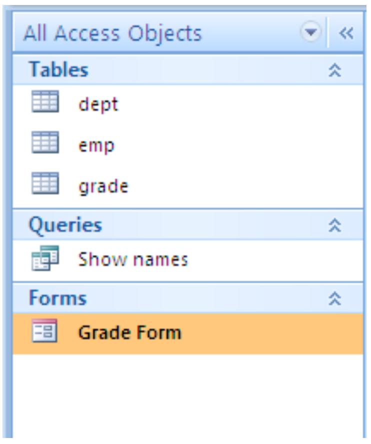
The default setting for the navigation pane shows the tables that exist (note that this may vary depending on the installation)



Click on the navigation pane menu bar to change the way that database objects are grouped and displayed – select *All Access Objects* to see the complete list subcategorised by type



Navigation pane showing 3 tables, 1 query and 1 form.



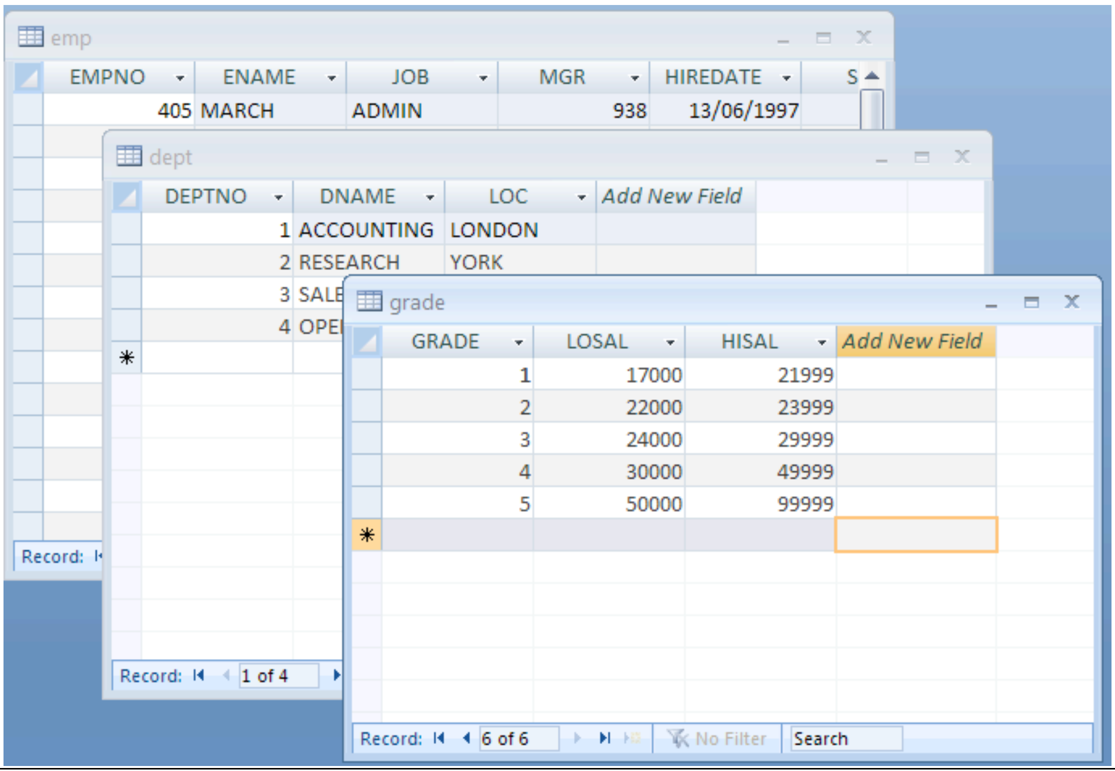
Activity 2: Ensure that you have All Access Objects selected in the navigation pane

Tables used on this course

The *session1* database contains three tables, EMP, DEPT and GRADE.

- EMP is a table containing employee information such as name, employment date, salary and employee number.
- DEPT is a table containing the names of each department.
- GRADE contains the salary scales for each salary grade.

Activity 3: To view these tables, double click on the table name in the navigation pane. Close the tables after examining them, but leave the database open.



EMPNO	ENAME	JOB	MGR	HIREDATE	S
405	MARCH	ADMIN	938	13/06/1997	

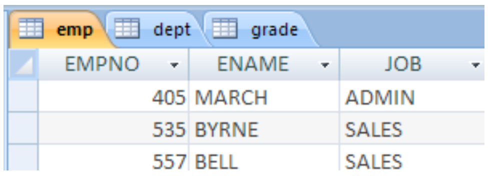
DEPTNO	DNAME	LOC
1	ACCOUNTING	LONDON
2	RESEARCH	YORK
3	SALE	
4	OPE	

Record: 1 of 4

GRADE	LOSAL	HISAL
1	17000	21999
2	22000	23999
3	24000	29999
4	30000	49999
5	50000	99999

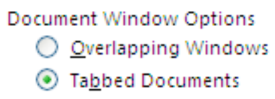
Record: 6 of 6

Depending on the initial set-up, Access may display multiple open database objects using tabs across the top of the objects.



EMPNO	ENAME	JOB
405	MARCH	ADMIN
535	BYRNE	SALES
557	BELL	SALES

To enable/disable this choice, click on the Office Button, select Access Options and then select/deselect Tabbed Documents. You may be prompted to reopen the database.



Document Window Options
Overlapping Windows [✓ Tabbed Documents]

The contents of the three tables in session 1 are reproduced here and will be useful when writing queries later.

EMP table

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
405	MARCH	ADMIN	938	13-Jun-97	18000		2
936	CASSY	ADMIN	734	23-Jul-02	19500		3
912	HAYES	ADMIN	824	04-Jun-01	21000		2
557	BELL	SALES	734	26-Mar-00	22500	500	3
690	AHMAD	SALES	734	05-Dec-97	22500	1400	3
970	BLACK	ADMIN	818	21-Nov-97	23000		1
880	TURNER	SALES	734	04-Jun-01	25000	0	3
535	BYRNE	SALES	734	15-Aug-97	26000	300	3
818	POLLARD	MANAGER	875	14-May-00	34500		1
734	COX	MANAGER	875	11-Jun-02	38500		3
602	BIRD	MANAGER	875	31-Oct-97	39750		2
824	REES	ANALYST	602	05-Mar-00	40000		2
938	GIBSON	ANALYST	602	05-Dec-97	40000		2
875	PARKER	PRESIDENT		09-Jul-02	60000		1

DEPT table

DEPTNO	DNAME	LOC
1	ACCOUNTING	LONDON
2	RESEARCH	YORK
3	SALES	BIRMINGHAM
4	OPERATIONS	LEEDS

GRADE table

GRADE	LOSAL	HISAL
1	17000	21999
2	22000	23999
3	24000	29999
4	30000	49999
5	50000	99999

Starting and Editing SQL

What is a Query?

A Query can be classified as a question that we require the database to provide an answer to. Example queries could be questions such as 'Who earns more than £23,000 per year?' or 'What will salaries look like if we give everyone a 5% pay increase?'

After creating the query, we execute it (or in Access terminology we *Run* it) and the database produces a response.

Queries can be edited and then re-executed if they are not quite correct.

Select queries allow us to view data without changing any of the core information.

- *Show me what would happen if we gave everyone a 5% increase*

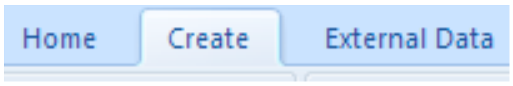
is different to

- *Give everyone a 5% increase*

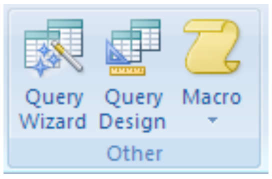
These queries (called *update* queries) will be covered later in the course.

The Access Query mechanism

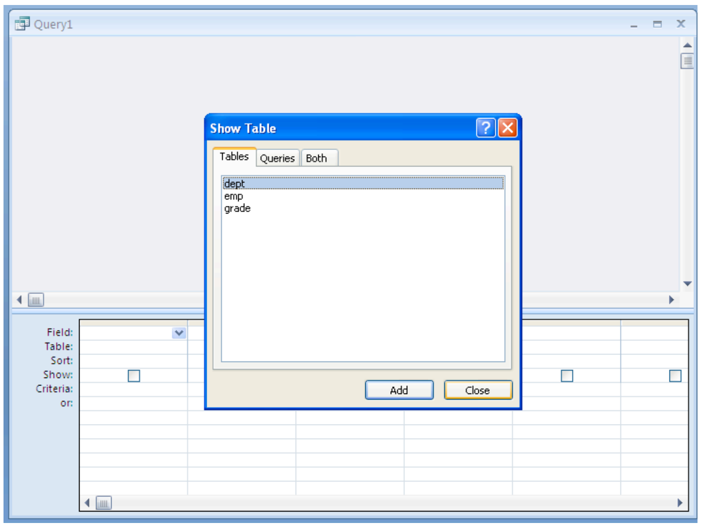
Clicking on the *Create* tab reveals icons to start a Query using either a wizard or the Query Design process



Clicking on Query Design will create a new (unnamed) blank query.



Do not add tables at this point as it is better to add them later (click *Close* in the Show Table dialog, without selecting any tables)



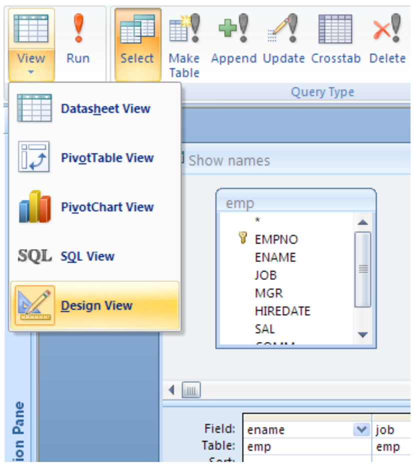
Activity 4: Create a new (blank) query by following the previous steps

Three Views

When using the Access Query mechanism five *views* of a query are available, through the *View* icon on the ribbon. Three of these are discussed here (note that only two are available initially). PivotTable View and PivotChart View are for specific types of queries.

The Design View

The design view allows query creation using a drag and drop principle. Although easier to use and learn, this interface is not transferable to other databases.

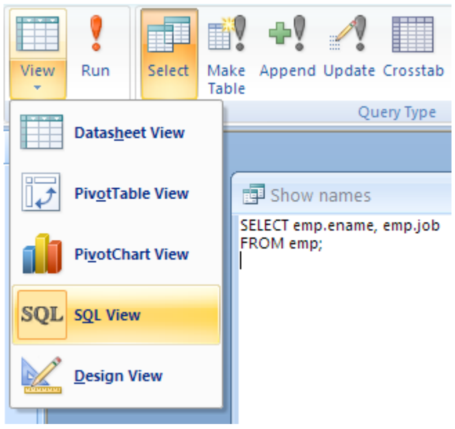


emp: *, EMPNO, ENAME, JOB, MGR, HIREDATE, SAL

Field:	ename	job
Table:	emp	emp

The SQL view

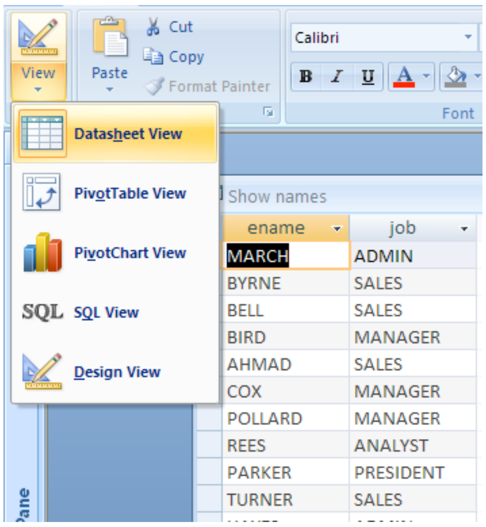
The SQL view allows the SQL source to be viewed in a primitive text editor. This is the principle way of entering and editing SQL text used on this course



```
SELECT emp.ename, emp.job
FROM emp;
```

The Datasheet view

The datasheet view actually 'runs' the query and generates the result. This is the same as selecting 'Run on a query through the Query manager interface on simple Select queries.



ename	job
MARCH	ADMIN
BYRNE	SALES
BELL	SALES
BIRD	MANAGER
AHMAD	SALES
COX	MANAGER
POLLARD	MANAGER
REES	ANALYST
PARKER	PRESIDENT
TURNER	SALES

Writing a query consists of the following phases:

- Write the query in *SQL view*
- View the query in Datasheet view
- Return to *SQL view* if changes are required

Activity 5: Create a simple query

Click on Create on the ribbon

Click on Query Design

Close the Show Table dialog (without selecting any tables)

Click on SQL from the View icon

In the simple text editor, type the following

```
select * from emp;
```

This syntax (which will be explained later) asks for everything from the *emp* table.

Click on *Datasheet view* from the *View* menu item to run the query.

Switch between Data Sheet and SQL view to see how Access switches modes.

Types of statements

SQL statements use the following reserved words:

Select	Used to retrieve data from the database, the most commonly used statement
Insert *Delete* *Update*	Used to enter, remove or change rows from a table. Together with Select, collectively known as the DML or Data Manipulation Language
Create *Alter* *Drop*	Used to set up, change or remove data structures such as tables, views or indexes. Collectively known as the DDL or Data Definition Language
Grant *Revoke* *Validate*	Used to give or remove access rights to data and data structures within an SQL database

Access does not fully implement the SQL standard, so not all of these are valid in Access queries. A *reserved word* is one that is specified as part of the core language and cannot be redefined by the user. Hence attempting to name a column 'select' will result in an error.

Projection in Relational Algebra and SQL

The Select statement

The Select statement is used to pull out and display information from a table. Its basic structure has this form:

```
SELECT select expression
FROM tables
```

In its simplest form, the select expression is a series of column names, separated by a comma.

`select ename,job`
`from emp;`
	`select deptno`
	`from emp;`
		`select ename,empno`
		`from emp;`

Select mirrors the Relational Algebra Projection operator (designated by a Π) which can be visualised as something that slices appropriate columns from a table. In Relational Algebra columns to be included in the result are generally listed (in subscript), before the relation name. The result of this operation is another relation / table.

$\Pi_{\text{ename, job}}$ (Emp)	Π_{deptno} (Emp)	$\Pi_{\text{ename,empno}}$ (Emp)
Gives the ename and job column	Gives all unique department numbers	Gives ename and empno from the emp relation

The result of a Relational Algebra operation will not contain duplicate rows (like in set theory). This is not quite the same as the SQL select operation as most databases list any duplicates entries.

The items after the *select* keyword can be one or more of the following, separated by commas:

	Example	*Explanation*
Column name	`empno`	Show the empno column
*	`*`	Show *all* columns
Arithmetic expression	`sal+1000`	Show salary + 1000
Character expression	`ename&job`	Combine two columns together
Arithmetic function	`round(sal, 2)`	Round salary to a certain number of places

There are other possible entries such as Date expressions, Aggregate functions, or actual values (character, numeric or date values). Date and character values must be given in single quotes (i.e. ‘Smith’ or ‘21-APR-86’). More information will be given on the date format later in the course.

Activity: The following examples demonstrate variations on the *Select* statement. Type these queries in and verify that they produce the results indicated. Note that you can only have one SQL statement per query in Access.

Examples

1: Display all columns and rows from table EMP

```
select * from emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
405	MARCH	ADMIN	938	13/06/1997	18000		2
535	BYRNE	SALES	734	15/08/1997	26000	300	3
557	BELL	SALES	734	26/03/2000	22500	500	3
602	BIRD	MANAGER	875	31/10/1997	39750		2
690	AHMAD	SALES	734	05/12/1997	22500	1400	3
734	COX	MANAGER	875	11/06/2002	38500		3
818	POLLARD	MANAGER	875	14/05/2000	34500		1
824	REES	ANALYST	602	05/03/2000	40000		2
875	PARKER	PRESIDENT		09/07/2002	60000		1
880	TURNER	SALES	734	04/06/2001	25000	0	3
912	HAYES	ADMIN	824	04/06/2001	21000		2
936	CASSY	ADMIN	734	23/07/2002	19500		3
938	GIBSON	ANALYST	602	05/12/1997	40000		2
970	BLACK	ADMIN	818	21/11/1997	23000		1

2: Display department numbers, employee names and their managers number.

```
select deptno,ename,mgr from emp;
```

deptno	ename	mgr
2	MARCH	938
3	BYRNE	734
3	BELL	734
2	BIRD	875
3	AHMAD	734
3	COX	875
1	POLLARD	875
2	REES	602
1	PARKER	
3	TURNER	734
2	HAYES	824
3	CASSY	734
2	GIBSON	602
1	BLACK	818

In Relational Algebra:

$\Pi_{\text{deptno, ename, mgr}}$ (emp)

3: Display all employees, along with the amount that a 50 % rise would make to the salary. Note that this doesn't change the value in the emp table.

```
select ename,(sal/100*50)+sal from emp;
```

ename	Expr1001
MARCH	27000
BYRNE	39000
BELL	33750
BIRD	59625
AHMAD	33750
COX	57750
POLLARD	51750
REES	60000
PARKER	90000
TURNER	37500
HAYES	31500
CASSY	29250
GIBSON	60000
BLACK	34500

In Relational Algebra:

$\Pi_{\text{ename, (sal/100*50)+sal}}$ (emp)

Note that Access generates a column name (as the mathematical expression `(sal/100*50)+sal` can't be used). To force a more sensible name to be used, use ' as columnname'.

Activity: Try `select ename,(sal/100*50)+sal as newsal from emp;`

4: Display all the unique department numbers.

```
select distinct deptno from emp;
```

deptno
1
2
3

In Relational Algebra:

Π_{deptno} (emp)

5: Concatenate and display the employee name and employee number in one column.

```
select empno&"-"&ename AS field1 from emp;
```

field1
405-MARCH
535-BYRNE
557-BELL
602-BIRD
690-AHMAD
734-COX
818-POLLARD
824-REES
875-PARKER
880-TURNER
912-HAYES
936-CASSY
938-GIBSON
970-BLACK

In Relational Algebra:

$\Pi_{\text{empno \& ”-“ \& ename}}$ (emp)

Concatenate means combine two pieces of data together into one. This operation is typically performed on text. Microsoft uses & to concatenate (as seen in Excel).

Exercises

In the following exercises, the query must be specified to produce the suggested result. There are spaces for you to write the SQL query in. Try and write down the equivalent Relational Algebra expression. Note that you may have to use the AS command to get correct column headings in SQL.

1. Display all the information in the table called grade

SQL:

GRADE	LOSAL	HISAL
1	17000	21999
2	22000	23999
3	24000	29999
4	30000	49999
5	50000	99999

In Relational Algebra:

2. Display the employee names and their hiredates

SQL:

ENAME	HIREDATE
MARCH	13/06/1997
BYRNE	15/08/1997
BELL	26/03/2000
BIRD	31/10/1997
AHMAD	05/12/1997
COX	11/06/2002
POLLARD	14/05/2000
REES	05/03/2000
PARKER	09/07/2002
TURNER	04/06/2001
HAYES	04/06/2001
CASSY	23/07/2002
GIBSON	05/12/1997
BLACK	21/11/1997

In Relational Algebra:

3. Display all the different types of jobs

SQL:

job
ADMIN
ANALYST
MANAGER
PRESIDENT
SALES

In Relational Algebra:

4. Display a list of employee names with monthly earnings (calculated from salary / 12). Note that you should call the resulting column *monthysal*

SQL:

ename	monthlysal
MARCH	1500
BYRNE	2166.6666667
BELL	1875
BIRD	3312.5
AHMAD	1875
COX	3208.3333333
POLLARD	2875
REES	3333.3333333
PARKER	5000
TURNER	2083.3333333
HAYES	1750
CASSY	1625
GIBSON	3333.3333333
BLACK	1916.6666667

In Relational Algebra:

5. Display a list of the locations where departments are located

SQL:

loc
LONDON
YORK
BIRMINGHAM
LEEDS

In Relational Algebra:

6. Display the grade, and the difference a 12% rise in the losal figure would make to each grade

SQL:

grade	increase
1	19040
2	24640
3	26880
4	33600
5	56000

In Relational Algebra:

7. Display the employee name and employee number as a single column separated by a hyphen, along with what difference a 5% increase in salary would make

SQL:

employ	newsal
405-MARCH	18900
535-BYRNE	27300
557-BELL	23625
602-BIRD	41737.5
690-AHMAD	23625
734-COX	40425
818-POLLARD	36225
824-REES	42000
875-PARKER	63000
880-TURNER	26250
912-HAYES	22050
936-CASSY	20475
938-GIBSON	42000
970-BLACK	24150

In Relational Algebra: