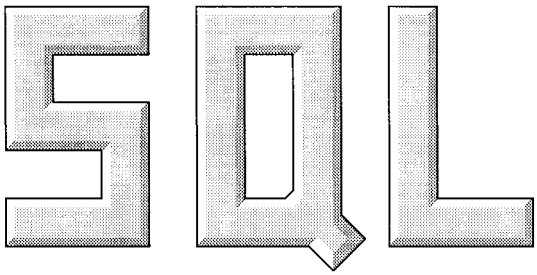


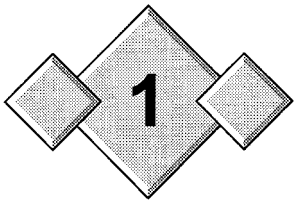
Structured Query Language

Version 1.2

Relational Systems Corporation
3646 Townline Road
Omer, Michigan 48749

Phone 989-653-3114 · Fax 989-653-3115

Table of Contents



INTRODUCTION

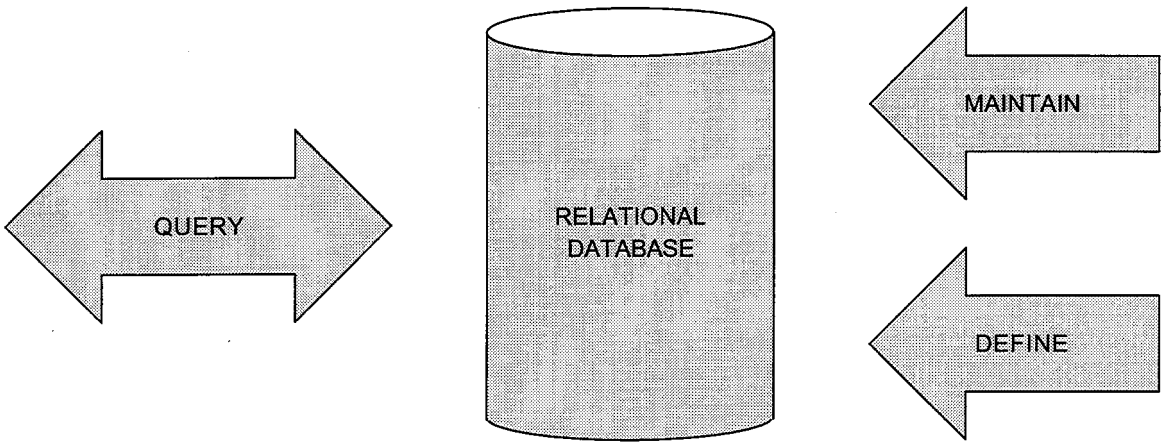
Course Objectives

What is a Relational Database?

What is SQL?

Course Objectives

When you finish this course, you will know what a relational database is, know what SQL is, and know what you can do to the former with the latter. You will also know why SQL is an important relational database language, and will actually be able to...



Query a relational database

The full range of SQL's query capabilities are explored in this course. Ample exercises, with answers, are provided for each topic. By the time you're done, you will be able to get information out of a relational database using SQL.

Maintain a relational database

You will also be able to put information into a relational database via SQL. You will learn how to insert new data, update existing data, and delete obsolete data. You will also understand how databases work in multi-user environments.

Define a relational database

Finally, you will know enough to actually create database objects with SQL, including, tables, constraints, indices, and views. Database design, however, which is covered in our other course offerings, is not addressed in this course.

What is a Relational Database?

A relational database is a collection of data stored in tables. Each table consists of one or more vertical columns, and zero or more horizontal rows. Pictured below are two tables from our exercise database, with three rows of sample data in each.

PERSONS

PERSON	NAME	BDATE	GENDER	COUNTRY	JOB
1	Einstein	1879-03-20	Male	Germany	S
2	Dickens	1812-07-22	Male	England	W
3	Dickinson	1830-07-15	Female	USA	W

COUNTRIES

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Germany	81337541	137823	1331000	German	100
USA	263814032	3679192	6380000	English	95
England	58295119	94251	980200	English	99

Tables are identified by name

Table names must be unique within a database. Typical SQL databases allow up to 18 characters in a table name. The first character must be a letter, while the remaining characters can be letters, numbers, or underscores.

Columns are identified by name

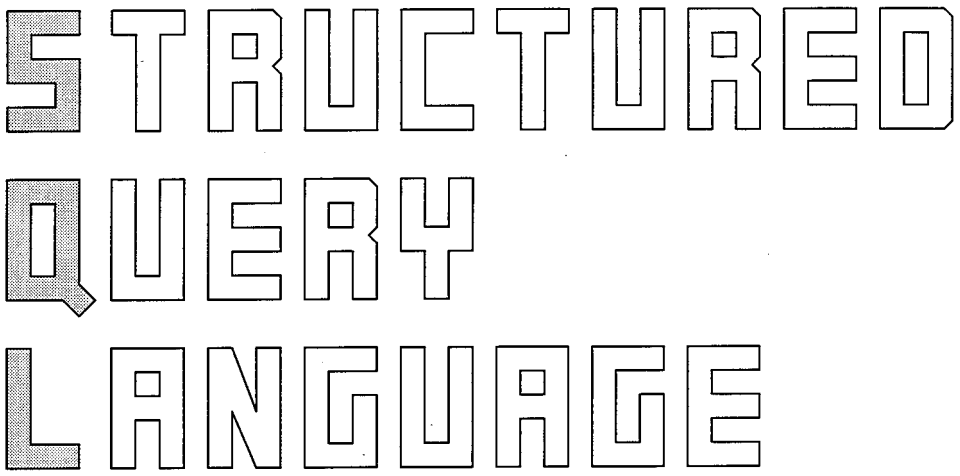
Column names are unique within each table, but the same column name can appear in different tables. Both tables above, for example, have COUNTRY columns.

Rows are identified by their contents

The rows of a table do not have names, nor is their position in a table fixed. Therefore, we refer to the rows of tables by describing the data values they contain: 'Person number 2,' for instance, or 'All English-speaking countries.'

What is SQL?

SQL is the most popular of all database languages. The name is an acronym for Structured Query Language, though SQL is much more than just a query language. Some pronounce the name as three separate letters, S-Q-L, but most say, 'sequel.'



Developed by IBM

Originally developed in 1974 by D. D. Chamberlin and others at IBM's San Jose Research Laboratory, SQL was finally released to the public as an integral part of IBM's SQL/DS and DB2 database products in 1982-1983.

Accepted by the world

SQL was adopted as an official standard by the American National Standards Institute (ANSI) in 1986, and in 1987 by the International Organization for Standardization (ISO). It is currently available in over 100 commercial products.

Implemented in many different ways

In spite of its popularity and its acceptance as an official standard, the SQL language is rather poorly defined and suffers from a number of internal inconsistencies. The result is a variety of SQL 'dialects' that differ slightly, but annoyingly, one from another.

What is SQL?

continued

The basic SQL language is actually quite small, and relatively easy to learn and use. It consists of six basic statement types (select, insert, update, delete, create, and drop) that can be conveniently grouped into the three categories shown below.

SQL STATEMENTS BY CATEGORY

CATEGORY	STATEMENT	PURPOSE
Query	SELECT	Display rows of one or more tables
Maintenance	INSERT	Add rows to a table
	UPDATE	Change rows in a table
	DELETE	Remove rows from a table
Definition	CREATE	Add tables, indices, views
	DROP	Remove tables, indices, views

Query statements

The SELECT statement, which has many different forms, is used to formulate all queries. SELECT statements 'ask questions' whose answers may be found in, or derived from, one or more of the tables in a database.

Maintenance statements

The INSERT statement, which has two different forms, is used to add new rows to a table (one or more at a time). The UPDATE statement is used to modify existing rows, while the DELETE statement is used to remove obsolete rows.

Definition statements

The CREATE statement, which also has several different forms, is used to define new tables, indices, and views in a database. Various forms of the DROP statement are used to remove these components when they are no longer required.

What is SQL?

A SQL statement is a collection of clauses, keywords, and parameters that perform a particular function. In the examples below, each clause is shown on a separate line; keywords appear in all uppercase letters, parameters in all lowercase letters.

QUERY STMT	MAINTENANCE STMT	DEFINITION STMT
SELECT columns FROM table WHERE comparisons	DELETE FROM table WHERE comparisons	CREATE INDEX index ON table (columns)

Clauses

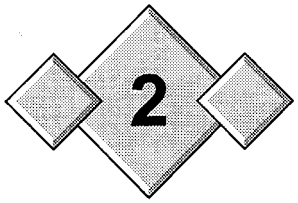
The various clauses of SQL statements are named after their initial words: the SELECT clause, for example, or the FROM clause. SQL allows multiple clauses per line, but most SQL programmers use separate lines for clarity and ease of editing.

Keywords

SQL reserves a small number of words, called keywords, for specific purposes. Keywords can be entered in upper, lower, or mixed-case letters; they are shown here in all uppercase. Keywords may not be used as table or column names.

Parameters

Parameters are the 'variable' parts of each clause. When formulating a SQL statement, you insert the appropriate column names, table names, and other such values in place of the lowercase parameters shown above.



BASIC QUERIES

Selecting All Columns & Rows

Selecting Specific Columns

Selecting Specific Rows

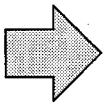
Sorting Rows

Eliminating Duplicate Rows

Selecting All Columns & Rows

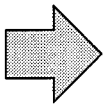
The simplest kind of query displays all the columns and all the rows of a single table. An asterisk is entered in the SELECT clause (to indicate that all columns should be included), and the table name is specified in the FROM clause, like so:

SQL STATEMENT
SELECT * FROM JOBS



JOB	TITLE
S	Scientist
E	Entertainer
W	Writer
I	Instructor

SQL STATEMENT
SELECT * FROM RELIGIONS



COUNTRY	RELIGION	PERCENT
Germany	Protestant	45
Germany	Catholic	37
England	Catholic	30
England	Anglican	70

All queries have SELECT and FROM clauses

The FROM clause follows the SELECT clause

The asterisk (*) means 'all columns'

The table name is specified in the FROM clause

Columns and rows appear in arbitrary order

Selecting All Columns & Rows continued

The general syntax of a SQL SELECT statement appears below. Keywords and examples are shown in uppercase letters, while parameters appear in lowercase.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
FROM	table	JOBS

Exercises

1. Ask the instructor for a brief tutorial in the use of the classroom database.

2. Select all the columns and all the rows of the PERSONS table.

3. Explore the COUNTRIES table using a SELECT statement.

4. Take a look at the ERRORS table.

5. Inspect the ARMIES table.

Extra Credit

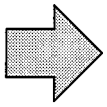
6. Most self-respecting databases have a table called SYSCOLUMNS. Take a peek.

7. Choose any other table from the SYSCOLUMNS table and display it.

Selecting Specific Columns

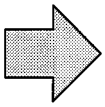
To select specific columns, we enter a list of columns in the SELECT statement (instead of an asterisk). Each column name is separated from the others by a comma and optional spaces. The columns are displayed in the order listed.

SQL STATEMENT
SELECT TITLE FROM JOBS



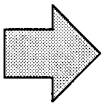
TITLE
Scientist
Entertainer
Writer
Instructor

SQL STATEMENT
SELECT TITLE, JOB FROM JOBS



TITLE	JOB
Scientist	S
Entertainer	E
Writer	W
Instructor	I

SQL STATEMENT
SELECT *, JOB FROM JOBS



JOB	TITLE	JOB
S	Scientist	S
E	Entertainer	E
W	Writer	W
I	Instructor	I

Column names are listed in the SELECT clause

Column names are separated by commas

Columns appear in the order listed

Rows appear in arbitrary order

Selecting Specific Columns

continued

The revised syntax of a SQL SELECT statement appears below. Note that it has been expanded to include the column list parameter in the SELECT clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
	col list	LANGUAGE, COUNTRY
FROM	table	JOBS

Exercises

1. Select only the JOB column from the JOBS table.

2. Show the NAME and BDATE columns from the PERSONS table.

3. The RELIGIONS table contains three columns — COUNTRY, RELIGION, and PERCENT. Display all of these columns, but show them in reverse order.

Extra Credit

4. Show all the columns of the COUNTRIES table, but put the COUNTRY column on both ends (so we can line 'em up with a ruler). Do this first without using an asterisk, then do it again with one.

Selecting Specific Rows

Rows in a table are identified by the values they contain. It is therefore important to understand the different categories of values that SQL supports, and the appropriate syntax for entering each kind of value in query statements.

VALUE SUMMARY

CATEGORY	DESCRIPTION	EXAMPLES
NUMERIC	positive values	3, +12
	negative values	-7, -1024000
	decimal values	3.141519, -.96
NON-NUMERIC	single words	'Chamberlin', 'SELECT'
	multiple words	'We love SQL', 'The LORD is good to me'
	single quotes	'10 O''Clock', 'I don''t know'
DATE	'yyyy-mm-dd' format	'1996-01-01', '1996-12-31'

Numeric values are entered in the normal way

Numeric values are specified in queries as one or more of the following characters:

+ - 0 1 2 3 4 5 6 7 8 9 .

The sign is optional, but must appear first. Only one decimal point is allowed. Note that commas, dollar signs, and percent signs are not allowed in numeric values.

Non-numeric values are enclosed in quotes

Non-numeric values, called strings, are entered inside of single quote marks. Use two consecutive single quote marks inside a string to represent a single quote.

Date values are entered in 'yyyy-mm-dd' format

While the SQL standard lacks a uniform representation of dates, the format shown above is supported in most SQL dialects. Dates must be enclosed in single quotes.

Selecting Specific Rows

continued

A comparison is a phrase that consists of a column name, a comparison operator, and a value. All comparisons yield a result of either true or false. Comparisons are used to specify which rows of a table should be included in the result of a query.

COMPARISON OPERATORS

OPERATOR	MEANING	EXAMPLE
=	Equal to	NAME = 'EINSTEIN'
<>	Not equal to	BDATE <> '1944-05-02'
<	Less than	POP < 100000
<=	Less than or equal to	NAME <= 'O"Grady'
>	Greater than	AREA > 999
>=	Greater than or equal to	BDATE >= '1962-06-19'

A column name is specified on the left

The column name can be entered in upper, lower, or mixed-case letters. In this text, columns names are shown in uppercase. Note that even though column names are non-numeric, they are not enclosed in quotes.

A value is specified on the right

Values must be entered appropriately for their value type — that is, numbers as they would normally be written, strings in single quotes, and dates in the 'yyyy-mm-dd' format. Values should be of the same type as columns to which they are compared.

An operator is specified in the middle

Comparison operators are placed between column names and values. Spaces on either side of a comparison operator are allowed, but are not required. However, spaces are not allowed between symbols in multiple symbol operators (<>, <=, >=).

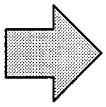
Selecting Specific Rows

continued

Comparisons are specified in the WHERE clause of a SQL SELECT statement. The WHERE clause must immediately follow the FROM clause. The result table includes all the rows of the source table where the comparison is true.

SQL STATEMENT

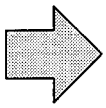
```
SELECT COUNTRY, AREA
FROM COUNTRIES
WHERE AREA > 3000000
```



COUNTRY	AREA
USA	3679192
Canada	3849674
China	3696100
Brazil	3286500
Russia	6592800

SQL STATEMENT

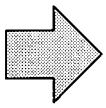
```
SELECT NAME, COUNTRY
FROM PERSONS
WHERE COUNTRY = 'RUSSIA'
```



NAME	COUNTRY
Rand	Russia
Tolstoy	Russia
Chekhov	Russia
Babel	Russia

SQL STATEMENT

```
SELECT NAME, BDATE
FROM PERSONS
WHERE BDATE < '1300-01-01'
```



NAME	BDATE
Dante	1265-03-21
Shikabu	1000-09-03
Augustino	0354-04-30
Magnus	1193-12-05
Paul	0013-06-01

The WHERE clause follows the FROM clause

Comparisons are entered in the WHERE clause

Rows where the comparison is true are displayed

Selecting Specific Rows

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the WHERE clause. A summary of SQL's comparison operators is also included.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000

OPERATOR	MEANING	EXAMPLE
=	Equal to	NAME = 'EINSTEIN'
<>	Not equal to	BDATE <> '1944-05-02'
<	Less than	POP < 100000
<=	Less than or equal to	NAME <= 'O"Grady'
>	Greater than	AREA > 999
>=	Greater than or equal to	BDATE >= '1962-06-19'

Exercises

1. Display all columns and rows in the COUNTRIES table. Now show only countries whose area is less than 30 square miles. There should be 5 rows in the result.

2. Show the name and country of everyone in the PERSONS table. Now limit the result to people who were born in Canada. The final result table contains 6 rows.

3. Show the name and birth date of people born after 1964-01-01. There are 9 rows.

Extra Credit

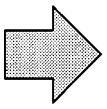
4. What is the birth date of the person named O'Toole?

Sorting Rows

To sort the rows of a result table, use the ORDER BY clause. This optional clause must be the last clause in a query. Column names, column positions in the SELECT clause, and the keyword DESC (descending) can be entered as parameters.

SQL STATEMENT

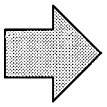
```
SELECT COUNTRY, AREA
FROM COUNTRIES
ORDER BY AREA
```



COUNTRY	AREA
Monaco	1
Vatican City	1
Nauru	8
Tuvalu	9
San Marino	24

SQL STATEMENT

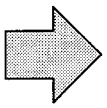
```
SELECT COUNTRY, AREA
FROM COUNTRIES
ORDER BY 2
```



COUNTRY	AREA
Monaco	1
Vatican City	1
Nauru	8
Tuvalu	9
San Marino	24

SQL STATEMENT

```
SELECT COUNTRY, AREA
FROM COUNTRIES
ORDER BY AREA DESC
```



COUNTRY	AREA
Russia	6592800
Canada	3849674
China	3696100
USA	3679192
Brazil	3286500

The ORDER BY clause must be last

A column name or position can be specified

DESC indicates a descending sort

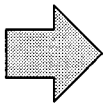
Sorting Rows

continued

Rows in a result table can be ordered based on the values in more than one column by entering a column list in the ORDER BY clause. The left-to-right order of the columns indicates the major-to-minor sort sequence.

SQL STATEMENT

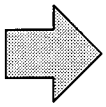
```
SELECT LANGUAGE, POP
FROM COUNTRIES
ORDER BY LANGUAGE
```



LANGUAGE	POP
Afrikaans	1651545
Albanian	3413904
Amharic	55979018
Arabic	549338
Arabic	65359623
Arabic	533916
Arabic	20643769

SQL STATEMENT

```
SELECT LANGUAGE, POP
FROM COUNTRIES
ORDER BY LANGUAGE, POP DESC
```



LANGUAGE	POP
Afrikaans	1651545
Albanian	3413904
Amharic	55979018
Arabic	65359623
Arabic	30120420
Arabic	29168848
Arabic	28539321

A column list is allowed in the ORDER BY clause

Column names are separated by commas

Columns are listed in major-to-minor sequence

Sorting Rows

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the ORDER BY clause. The square brackets indicate that DESC is optional.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Show all columns of the PERSONS table for people from Ireland. There are 6 such people. Now show them in alphabetical order.

2. Display all columns of the COUNTRIES table for German-speaking countries. The result table contains 4 rows. Now sort by GNP, largest on top.

3. Display the country, job, and name of all Italians in the database, sorted by name within job. There are 10 rows in the result table.

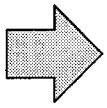
Extra Credit

4. Which country has the largest military budget? Which has the smallest? Which has the most troops? the most tanks? ships? planes?

Eliminating Duplicate Rows

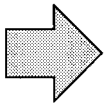
Typically, tables do not contain duplicate rows. Queries, however, may result in duplicate rows when a subset of the columns is selected. The keyword DISTINCT is used in the SELECT clause to prevent the display of duplicate rows in a result table.

SQL STATEMENT
SELECT RELIGION FROM RELIGIONS WHERE PERCENT = 100 ORDER BY RELIGION



RELIGION
Catholic
Catholic
Catholic
Catholic
Eastern Or...
Lutheran
Muslim
Muslim
Muslim
Sunni Mus...

SQL STATEMENT
SELECT DISTINCT RELIGION FROM RELIGIONS WHERE PERCENT = 100 ORDER BY RELIGION



RELIGION
Catholic
Eastern Or...
Lutheran
Muslim
Sunni Mus...

DISTINCT is entered after the keyword SELECT

DISTINCT eliminates duplicate rows

DISTINCT may only be entered once in a query

Eliminating Duplicate Rows

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the optional keyword DISTINCT in the SELECT clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

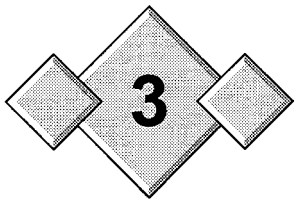
1. Display just the JOB column from the PERSONS table. There are 402 rows (don't bother counting them). Now eliminate the duplicates. Only 7 should remain.

2. Display an alphabetical list of languages where the literacy rate is less than 30 percent. Now eliminate the duplicates. The final result should have 9 rows.

3. List the countries that have produced at least one scientist, sorted by country. Now remove duplicates. Only 10 rows should remain.

Extra Credit

4. Make an alphabetical list of religions practiced by less than five percent of their country's population. Eliminate any duplicate rows. 7 rows should remain.



ADVANCED OPERATORS

The LIKE Operator

The AND Operator

The BETWEEN Operator

The OR Operator

The IN Operator

The IS NULL Operator

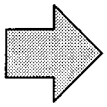
Precedence and Negation

The LIKE Operator

The LIKE operator is used to find values that match a pattern. Patterns are always entered in quotes. A percent symbol is used to represent zero or more unknown characters; an underscore represents a single unknown character.

SQL STATEMENT

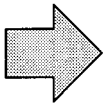
```
SELECT NAME, COUNTRY
FROM PERSONS
WHERE NAME LIKE 'Z%'
```



NAME	COUNTRY
Zola	France
Zimbalist	USA
Zwingli	Sweden

SQL STATEMENT

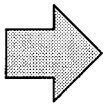
```
SELECT NAME, COUNTRY
FROM PERSONS
WHERE NAME LIKE 'EINST_ _N' ;
```



NAME	COUNTRY
Einstein	Germany

SQL STATEMENT

```
SELECT NAME, COUNTRY
FROM PERSONS
WHERE NAME LIKE '%ST_ _N%'
```



NAME	COUNTRY
Einstein	Germany
Springsteen	USA
Steinbeck	USA
Silverstein	USA

LIKE finds values that match a pattern

Percent (%) represents zero or more characters

Underscore (_) represents one character

The LIKE Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the LIKE operator in the WHERE clause, and the % and _ characters in the example.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST_ _N%'
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Select all countries that include 'guinea' in their name. You should find 4 rows.

2. Display all columns for people with the letter z as the second character in their name. There are 2 such individuals.

3. Show all columns for persons born on July 15. There are 6 such people.

Extra Credit

4. Find all people with an apostrophe in their name. There are 3 such people.

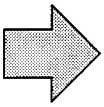
5. Make a list of religions that contain the word 'orthodox' in them. Sort the list, and make sure there are no duplicates. The result contains 10 rows.

The AND Operator

The AND operator is used to combine two comparisons, creating a compound comparison. The keyword AND is placed between the two comparisons. Both comparisons must evaluate to true for the compound comparison to be true.

SQL STATEMENT

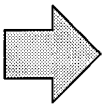
```
SELECT NAME, BDATE
FROM PERSONS
WHERE NAME LIKE 'A%'
AND BDATE >= '1900-01-01'
```



NAME	BDATE
Anne	1950-08-15
Albert II	1934-06-06
Achebe	1930-11-16
Archer	1947-08-25
Azimov	1920-08-22
Andrews	1935-10-01

SQL STATEMENT

```
SELECT COUNTRY, GNP
FROM COUNTRIES
WHERE GNP >= 1000
AND GNP <= 2000
```



COUNTRY	GNP
Eritrea	1700
Guyana	1400
Jamaica	1500
Suriname	1170

AND combines two comparisons

AND is placed between two comparisons

Both comparisons must evaluate to true

The AND Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the AND operator in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. List all columns for scientists from Germany. You should find 7 scientists.

2. Display all columns for countries with a GNP less than three billion, and a literacy rate less than 40 percent. Remember that GNP is stored in millions of dollars. Sort the result by GNP. The result table contains 7 rows.

3. Show the country name and literacy rate of countries where the literacy rate is between 55 and 60 percent. There are 7 rows.

Extra Credit

4. Show all columns for English people born in the 16th century, sorted by name. The result table contains 4 rows.

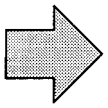
5. Show all columns for the eight armies that have more than 100,000 troops, less than 1,000 tanks, less than 1,000 ships, and less than 1,000 planes.

The BETWEEN Operator

Some compound comparisons using the AND operator can be more conveniently expressed using the BETWEEN operator. BETWEEN compares each column value with a range of values. The range always includes the end points.

SQL STATEMENT

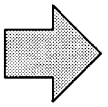
```
SELECT COUNTRY, LITERACY
FROM COUNTRIES
WHERE LITERACY >= 55
AND LITERACY <= 60
ORDER BY LITERACY
```



COUNTRY	LITERACY
Equatorial...	55
Guatemala	55
Congo	57
Algeria	57
Ghana	60
Iraq	60
Palau	60

SQL STATEMENT

```
SELECT COUNTRY, LITERACY
FROM COUNTRIES
WHERE LITERACY BETWEEN 55 AND 60
ORDER BY LITERACY
```



COUNTRY	LITERACY
Equatorial...	55
Guatemala	55
Congo	57
Algeria	57
Ghana	60
Iraq	60
Palau	60

BETWEEN compares each column value to a range

The range includes both end points

The second value must be greater than the first

The BETWEEN Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the BETWEEN operator in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Using the BETWEEN operator, list all columns for countries with areas greater than or equal to five and less than or equal to 75 square miles. There are 5 rows.

2. Select all columns for countries having between 100,000 and 200,000 people. There are 6 rows in the result.

3. Select all columns for armies whose equipment includes over 300 but less than 400 planes. There are 9 such armies.

Extra Credit

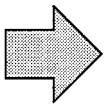
4. Select all columns for armies with at least 300,000 troops, whose budget is between 10 and 100 billion dollars. Remember that BUDGET is in millions of dollars. There are 4 such armies.

The OR Operator

The OR operator is used to combine two comparisons, creating a compound comparison. The keyword OR is placed between the two comparisons. Either or both comparisons must evaluate to true for the compound comparison to be true.

SQL STATEMENT

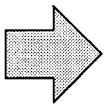
```
SELECT NAME, BDATE
FROM PERSONS
WHERE NAME = 'LUTHER'
OR NAME = 'CALVIN'
```



NAME	BDATE
Luther	1483-06-15
Calvin	1509-06-04

SQL STATEMENT

```
SELECT COUNTRY, LANGUAGE
FROM COUNTRIES
WHERE COUNTRY = 'GHANA'
OR COUNTRY = 'USA'
OR COUNTRY = 'FIJI'
```



COUNTRY	LANGUAGE
Fiji	English
USA	English
Ghana	English

OR combines two comparisons

OR is placed between two comparisons

Either or both comparisons must evaluate to true

The OR Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the OR operator in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
	cmpr OR cmpr	NAME = 'LUTHER' OR NAME = 'CALVIN'
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Display all columns in the ARMIES table for Israel and Iraq.

2. Show all columns for people who were born on either 1835-10-19 or 1917-02-06. There are 4 such people... or are there?

3. List names and birth dates for Poe, Hugo, and Dahl.

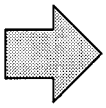
Extra Credit

4. How many people in the database were born in the 13th or 15th century?

The IN Operator

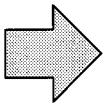
Some compound comparisons using the OR operator can be more conveniently expressed using the IN operator. IN compares each column value with a list of values. The list is enclosed in parentheses; the values are separated with commas.

SQL STATEMENT
SELECT NAME, BDATE FROM PERSONS WHERE NAME = 'POE' OR NAME = 'HUGO' OR NAME = 'DAHL'



NAME	BDATE
Hugo	1802-08-05
Dahl	1916-09-01
Poe	1809-04-09

SQL STATEMENT
SELECT NAME, BDATE FROM PERSONS WHERE NAME IN ('POE', 'HUGO', 'DAHL')



NAME	BDATE
Hugo	1802-08-05
Dahl	1916-09-01
Poe	1809-04-09

IN compares each column value to a list

The list is enclosed in parentheses

Values in the list are separated by commas

The IN Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the IN operator in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
	cmpr OR cmpr	NAME = 'LUTHER' OR NAME = 'CALVIN'
	col IN (value list)	NAME IN ('POE', 'HUGO', 'DAHL')
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Show all columns for Einstein, Galilei, and Newton, using IN.

2. Display all columns for countries with a literacy rate of 20, 40, or 60 percent. There are 5 such countries.

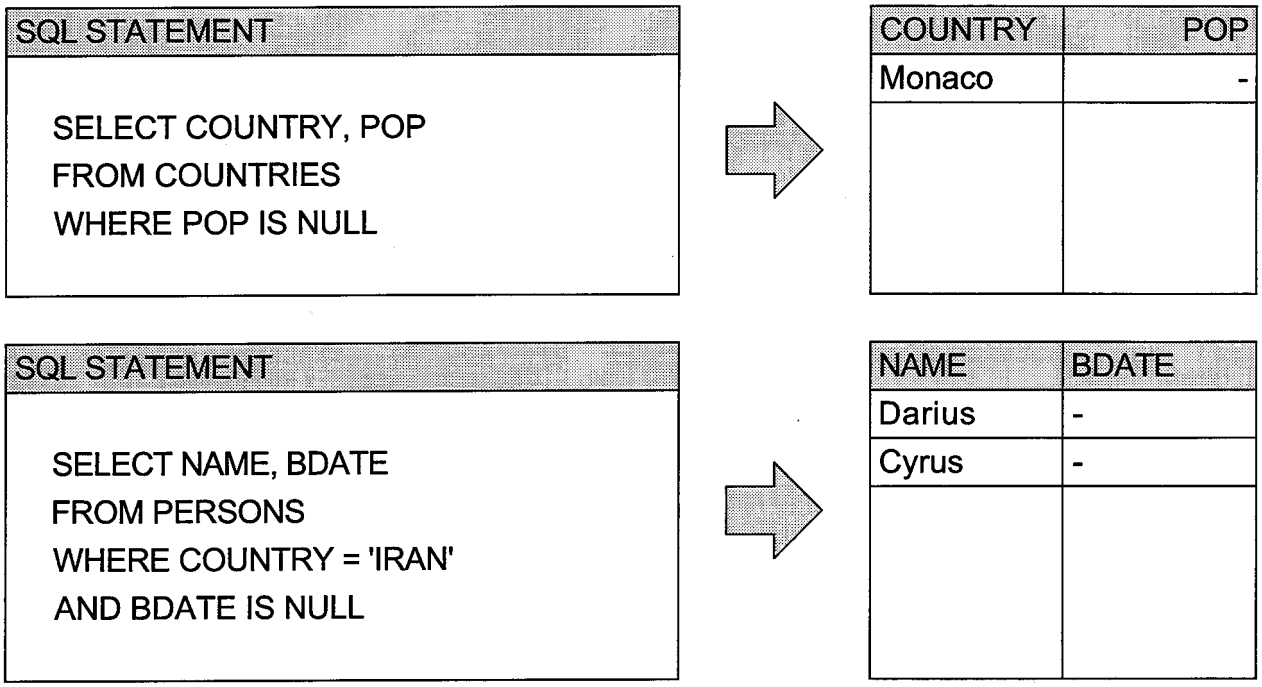
3. List the names and countries of all scientists in the database who are from Germany, Austria, or Italy. There are 11 rows in the result table.

Extra Credit

4. Find all people from Germany, Austria, or Italy who are writers, entertainers, or business leaders. There are 11. Official job identifiers are in the JOBS table.

The IS NULL Operator

A null value is a missing entry in a column. Null means 'unknown' or 'does not apply.' Nulls are neither blanks nor zeros (two nulls are not necessarily equal, and you cannot do arithmetic with nulls). The IS NULL operator locates rows with null values.



Nulls are missing values

Nulls are not the same as blanks

Nulls are not the same as zeros

IS NULL locates null values

The IS NULL Operator

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the IS NULL operator in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
	cmpr OR cmpr	NAME = 'LUTHER' OR NAME = 'CALVIN'
	col IN (value list)	NAME IN ('POE', 'HUGO', 'DAHL')
	col IS NULL	POP IS NULL
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

Exercises

1. Show all columns for countries whose GNP is unknown. There is 1 such country.

2. Display all columns for entertainers whose gender is null. There are 2 such.

3. Select all columns for Israelis whose birth date is unknown. There are 9 of them.

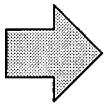
Extra Credit

4. How many countries have an area less than 10 square miles and a GNP greater than 250 million? How many countries have an area less than 10 square miles and a GNP less than or equal to 250 million? Given those results, how many countries would you guess have an area less than 10 square miles?

Precedence and Negation

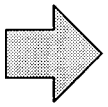
Parentheses are used to indicate precedence — the order in which comparisons are evaluated. SQL evaluates comparisons enclosed in parentheses first. The keyword NOT is used to negate, or reverse, the result of a comparison.

SQL STATEMENT
SELECT JOB, NAME FROM PERSONS WHERE COUNTRY = 'ITALY' AND (JOB = 'S' OR JOB = 'W') ORDER BY JOB, NAME



JOB	NAME
S	Avogadro
S	Fermi
S	Galilei
W	Boccaccio
W	Dante
W	Petrarca

SQL STATEMENT
SELECT JOB, NAME FROM PERSONS WHERE COUNTRY = 'ITALY' AND NOT (JOB = 'S' OR JOB = 'W') ORDER BY JOB, NAME



JOB	NAME
E	Fabio
M	Epiphani
M	Ptolemio
T	Augustino

Comparisons in parentheses are evaluated first

NOT is placed in front of a comparison

NOT reverses the result of a comparison

Precedence and Negation

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of parentheses and the keyword NOT in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col list	LANGUAGE, COUNTRY
FROM	table	JOBS
WHERE	col oper value	AREA > 3000000
[NOT]	col LIKE pattern	NAME LIKE '%ST__N%'
	cmpr AND cmpr	NAME LIKE '%ST__N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
	cmpr OR cmpr	NAME = 'LUTHER' OR NAME = 'CALVIN'
	col IN (value list)	NAME IN ('POE', 'HUGO', 'DAHL')
	col IS NULL	POP IS NULL
	(compound cmpr)	(JOB = 'S' OR JOB = 'W')
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

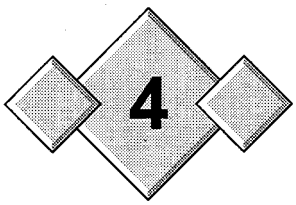
Exercises

1. Display all columns for Germans who are either theologians or business leaders. Sort the result by name. There are 3 rows.

2. Modify the previous query to find all Germans who are not business leaders or theologians. The result contains 11 rows.

Extra Credit

3. How many armies, not including the USA and Russia, have a military budget greater than 30 billion dollars? There are...



EXPRESSIONS

Arithmetic Expressions

Expressions in Other Clauses

Column Aliases

Arithmetic Expressions

An arithmetic expression is a phrase formed with operands (numeric values and/or column names) and arithmetic operators (shown below). Arithmetic expressions are evaluated by SQL and replaced with the appropriate numeric value.

ARITHMETIC OPERATORS

OPERATOR	MEANING	EXAMPLE
+	Add	2 + 2
-	Subtract	BDATE - 365
*	Multiply	POP * 1.25
/	Divide	PERCENT / 100
( )	Precedence	2 + (4 / 2)

Operands can be numeric values or column names

In an expression, valid operands include numbers, column names, and other expressions. When a column is used in an expression, the values in that column must be numeric or valid dates. Only addition and subtraction are valid with dates.

Operators are specified between operands

An arithmetic operator is placed between its operands, in the usual way. While spaces are not required on either side of arithmetic operators, spaces are included in the examples above for readability.

Expressions in parentheses are evaluated first

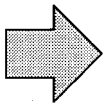
Arithmetic expressions that are enclosed in parentheses are evaluated by SQL first. The results are then combined with other expressions in the same statement. The placement of parentheses can significantly affect the value of the expression.

Arithmetic Expressions

continued

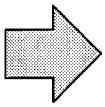
Along with the asterisk and column names, expressions can be specified in the column list of a SELECT clause. SQL inserts a calculated column in the result table at the given position. The new column name defaults to the expression itself.

SQL STATEMENT
SELECT COUNTRY, POP / AREA FROM COUNTRIES WHERE LANGUAGE = 'GERMAN' ORDER BY COUNTRY



COUNTRY	POP/AREA
Austria	246.66
Germany	590.15
Liechtenst...	494.41
Switzerland	444.47

SQL STATEMENT
SELECT COUNTRY, GNP * 1.1 FROM COUNTRIES WHERE LANGUAGE = 'GERMAN' ORDER BY COUNTRY



COUNTRY	GNP*1.1
Austria	147840
Germany	1464100
Liechtenst...	693
Switzerland	164010

Expressions are specified in the SELECT clause

Expressions produce new columns in the result

Expressions are calculated for each row

Arithmetic Expressions

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of expressions in the SELECT clause. The arithmetic operators are also summarized.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col list	LANGUAGE, COUNTRY
	expr list	POP / AREA
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3

OPERATOR	MEANING	EXAMPLE
+	Add	2 + 2
-	Subtract	BDATE - 365
*	Multiply	POP * 1.25
/	Divide	PERCENT / 100
( )	Precedence	2 + (4 / 2)

Exercises

1. If 20% of Canadians moved to the USA, how many new Americans would that be?

2. How much money is spent per trooper in the USA? in China?

3. Calculate the total number of military vehicles (tanks plus ships plus planes) owned by the United States.

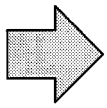
Extra Credit

4. 45% of our military budget is spent on $125 hammers. How many do we own?

Expressions in Other Clauses

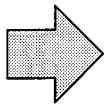
Since SQL treats expressions in SELECT statements as 'virtual columns' filled with calculated values, we can use expressions in place of column names in both the WHERE and ORDER BY clauses, as illustrated in the examples below.

SQL STATEMENT
SELECT COUNTRY, POP / AREA FROM COUNTRIES WHERE POP / AREA > 2000



COUNTRY	POP/AREA
Maldives	2272.26
Malta	3029.58
Singapore	11702.29
Bahrain	2148.97
Bangladesh	2235.70

SQL STATEMENT
SELECT COUNTRY, POP / AREA FROM COUNTRIES WHERE POP / AREA > 2000 ORDER BY POP / AREA DESC



COUNTRY	POP/AREA
Singapore	11702.29
Malta	3029.58
Maldives	2272.26
Bangladesh	2235.70
Bahrain	2148.97

Expressions can be used in the WHERE clause

Expressions can be used in the ORDER BY clause

Expressions in Other Clauses continued

The current syntax of a SQL SELECT statement appears below. Note the addition of expressions in the WHERE and ORDER BY clauses.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col list	LANGUAGE, COUNTRY
	expr list	POP / AREA
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
	expr oper value	POP / AREA > 300
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC

Exercises

1. Display the country, population, area, and the population density (POP/AREA) for countries whose population density is less than seven people per square mile. Put the highest population density on top. The result table contains 7 rows.

2. Show the country, population, literacy rate, and number of literate people for all countries with over 100 million literates. To find the number of literates, divide the literacy rate by 100 and multiply the result by population. There are 7 rows.

Extra Credit

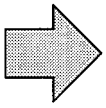
3. Let's assume that all armies spend 30 percent of their budget on ashtrays, and that the average military ashtray costs $650. Display each country and the number of ashtrays per soldier, but only if the number of ashtrays per soldier is greater than 10. Put the largest count on top. There are 9 such countries.

Column Aliases

The keyword AS can be used in the SELECT clause to define a column alias — a user-assigned name for a column. Column aliases can be specified for any column, but they are most frequently used to give meaningful names to calculated columns.

SQL STATEMENT

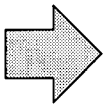
```
SELECT COUNTRY, POP / AREA
FROM COUNTRIES
WHERE POP / AREA > 2000
ORDER BY POP / AREA DESC
```



COUNTRY	POP/AREA
Singapore	11702.29
Malta	3029.58
Maldives	2272.26
Bangladesh	2235.70
Bahrain	2148.97

SQL STATEMENT

```
SELECT COUNTRY,
POP / AREA AS DENSITY
FROM COUNTRIES
WHERE DENSITY > 2000
ORDER BY DENSITY DESC
```



COUNTRY	DENSITY
Singapore	11702.29
Malta	3029.58
Maldives	2272.26
Bangladesh	2235.70
Bahrain	2148.97

Column aliases are defined in the SELECT clause

Aliases can be used in the WHERE clause

Aliases can be used in the ORDER BY clause

AS is optional in some dialects

Column Aliases

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the keyword AS and column aliases in the SELECT clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	LANGUAGE AS LANG, COUNTRY
	expr [AS alias] list	POP / AREA AS DENSITY
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
	expr oper value	POP / AREA > 300
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC

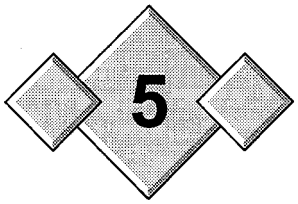
Exercises

1. Show the country, population, literacy rate, and number of literate people for all countries with over 100 million literates. Call the calculated column READERS, and use this alias wherever appropriate. There are 7 rows in the result.

2. Using a column alias, display the country, population, GNP, and GPP (short for gross personal product = GNP * 1000000 / POP) for countries whose GPP is over 20,000. Sort by GPP descending. The result contains 9 rows.

Extra Credit

3. It is proposed that the nations of the world reduce their troops by 20% and apply the savings to nuclear power plants. An average trooper costs $20,000. How much will each country contribute? Call the calculated column FISSION_FUND, put the largest on top, and eliminate countries contributing less than two billion.



FUNCTIONS

Statistical Functions

Grouping

Functions in Other Clauses

Statistical Functions

A statistical function is a built-in program that accepts a parameter and returns a summary value. The parameter may be either a column name or an expression. The five statistical functions supported by standard SQL are shown in the following table.

STATISTICAL FUNCTIONS

FUNCTION	MEANING	EXAMPLE
COUNT ( )	Count all rows	COUNT (*)
	Count non-null rows	COUNT (JOB)
	Count unique rows	COUNT (DISTINCT JOB)
SUM ( )	Total value	SUM (POP)
MIN ( )	Smallest value	MIN (POP)
MAX ( )	Largest value	MAX (POP)
AVG ( )	Average value	AVG (POP / AREA)

Statistical functions accept parameters

Parameters are either column names or expressions. Parameters must always be enclosed in parentheses. Note that spaces inside and outside the parentheses are optional, but are shown in the examples above for readability.

Statistical functions return summary values

SUM and AVG can only be used with columns that contain numeric values. COUNT, MIN, and MAX can be used with any type of column. All statistical functions operate on a single column or expression.

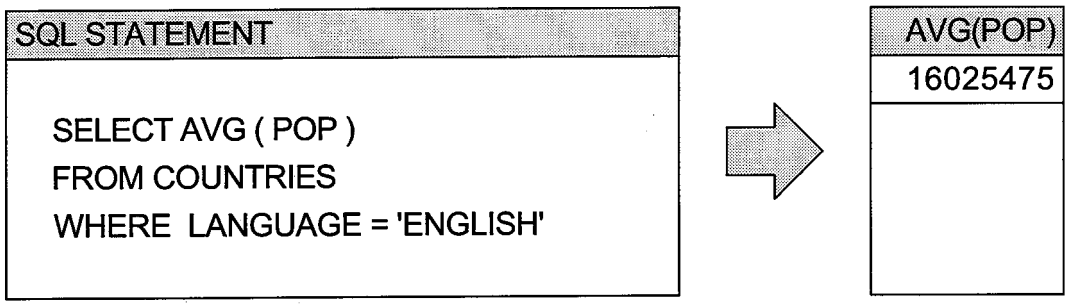
COUNT accepts a variety of parameters

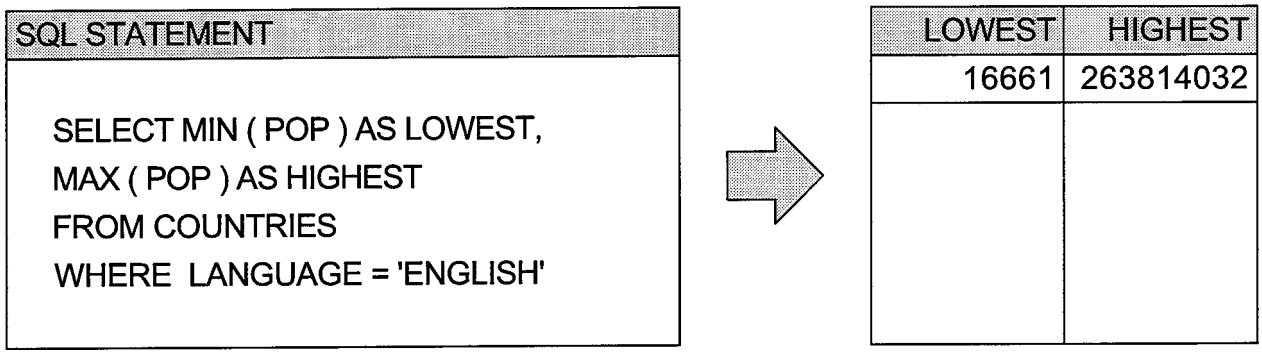
COUNT(*) produces a count of rows. COUNT(column) produces a count of rows where the specified column contains non-null values. COUNT(DISTINCT column) produces a count of unique, non-null values in the given column.

Statistical Functions

continued

If a SELECT clause contains nothing but statistical functions, SQL displays grand totals for the query. The resulting table contains one row, with one column for each statistical function. Column aliases may be used to rename the column(s).





Only functions are specified in the SELECT clause

The result contains only one row

The result contains one column for each function

Column aliases may be assigned

Statistical Functions

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of functions in the SELECT clause. A summary of statistical functions is also shown.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	LANGUAGE AS LANG, COUNTRY
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC

FUNCTION	MEANING	EXAMPLE
COUNT ( )	Count all rows	COUNT (*)
	Count non-null rows	COUNT (JOB)
	Count unique rows	COUNT (DISTINCT JOB)
SUM ( )	Total value	SUM (POP)
MIN ( )	Smallest value	MIN (POP)
MAX ( )	Largest value	MAX (POP)
AVG ( )	Average value	AVG (POP / AREA)

Exercises

1. Find the total number of troops in the ARMIES table. You should get 17,846,400.

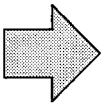
2. Show the minimum, maximum, and average literacy rates for French-speakers. The minimum is 18%, the maximum is 100%, and the average is 51.38%.

3. Find a count of countries (190) and a count of distinct languages (79) in one query.

Grouping

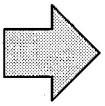
If a SELECT clause contains column names and functions, SQL displays subtotals. The specified columns must also be listed in the GROUP BY clause. SQL divides the table into groups, calculates subtotals for each, and displays one row per group.

SQL STATEMENT
SELECT JOB, COUNT (*) AS TOTAL FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB ORDER BY JOB



JOB	TOTAL
B	28
E	113
M	36
R	14
S	28
T	13
W	99

SQL STATEMENT
SELECT JOB, COUNTRY, COUNT (*) AS TOTAL FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB, COUNTRY ORDER BY JOB, COUNTRY



JOB	COUNTRY	TOTAL
B	England	1
B	France	1
B	Germany	2
B	USA	24
E	Austria	1
E	Belgium	1
E	Canada	5
E	England	14

Group columns are specified in the SELECT clause

Group columns are repeated in the GROUP BY clause

GROUP BY follows the FROM and WHERE clauses

Grouping

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the GROUP BY clause following the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col [AS alias] list	JOB, COUNTRY
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
GROUP BY	col list	JOB, COUNTRY
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC

Exercises

1. Display each language and the total number of people that speak it. Limit your result to the following languages: Hebrew, Spanish, English, and French.

2. List the minimum, maximum, and average literacy rates for the above languages.

3. In a single query, calculate the number of males and females in each of the following countries: Canada and France. Sort by gender within country.

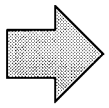
Extra Credit

4. List the number of people of each gender that do each job. Do not include people whose gender is unknown. Sort the result by gender within job. The result table should contain 12 rows.

Functions in Other Clauses

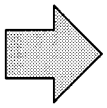
Since SQL sorts a result after all other processing has taken place, a function may be specified in the ORDER BY clause, as shown below. The function can be entered directly, or referenced via a column alias defined in the SELECT clause.

SQL STATEMENT
SELECT JOB, COUNT (*) FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB ORDER BY COUNT (*) DESC



JOB	COUNT(*)
E	113
W	99
M	36
B	28
S	28
R	14
T	13

SQL STATEMENT
SELECT JOB, COUNT (*) AS TOTAL FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB ORDER BY TOTAL DESC



JOB	TOTAL
E	113
W	99
M	36
B	28
S	28
R	14
T	13

Functions can be used in the ORDER BY clause

Functions can be entered directly

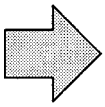
Functions can be referenced via column aliases

Functions in Other Clauses

continued

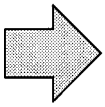
Functions cannot be specified in the WHERE clause because that clause is evaluated before grouping (and function execution) takes place. Functions can, however, appear in the HAVING clause, which is processed after grouping occurs.

SQL STATEMENT
SELECT JOB, COUNT (*) AS TOTAL FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB ORDER BY TOTAL DESC



JOB	TOTAL
E	113
W	99
M	36
B	28
S	28
R	14
T	13

SQL STATEMENT
SELECT JOB, COUNT (*) AS TOTAL FROM PERSONS WHERE GENDER = 'MALE' GROUP BY JOB HAVING TOTAL > 30 ORDER BY TOTAL DESC



JOB	TOTAL
E	113
W	99
M	36

The HAVING clause follows the GROUP BY clause

The HAVING clause is just like WHERE except...

The HAVING clause is executed after grouping

Functions in Other Clauses

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the HAVING clause, and the extension of the ORDER BY clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	LANGUAGE AS LANG, COUNTRY
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table	JOBS
WHERE	comparisons	AREA > 3000000
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC
	func [DESC] list	COUNT (*) DESC

Exercises

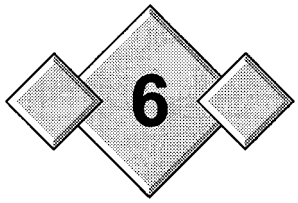
1. Count the writers in each country. Sort by count descending. There are 12 rows.

2. List languages that are spoken by less than a million people. There are 11 rows.

3. Find the minimum, maximum, and average literacy rates for each language whose minimum is not equal to the maximum. The result contains 13 rows.

Extra Credit

4. Display language, total area, and total population for all languages spoken in more than one country. Put the largest total area on top. There are 13 rows.



MULTI-TABLE QUERIES

Joins

Table Aliases

Unions

Joins

A full column name consists of a table name, a period, and a column name, with no intervening spaces. Full column names are necessary in multi-table queries when the same column name appears in more than one table.

FULL COLUMN NAMES FOR THE EXERCISE DATABASE

TABLE	COLUMN	FULL COLUMN NAME
PERSONS	PERSON	PERSONS.PERSON
	NAME	PERSONS.NAME
	BDATE	PERSONS.BDATE
	GENDER	PERSONS.GENDER
	COUNTRY	PERSONS.COUNTRY
	JOB	PERSONS.JOB
COUNTRIES	COUNTRY	COUNTRIES.COUNTRY
	POP	COUNTRIES.POP
	AREA	COUNTRIES.AREA
	GNP	COUNTRIES.GNP
	LANGUAGE	COUNTRIES.LANGUAGE
	LITERACY	COUNTRIES.LITERACY
ARMIES	COUNTRY	ARMIES.COUNTRY
	BUDGET	ARMIES.BUDGET
	TROOPS	ARMIES.TROOPS
	TANKS	ARMIES.TANKS
	SHIPS	ARMIES.SHIPS
	PLANES	ARMIES.PLANES
JOBS	JOB	JOBS.JOB
	TITLE	JOBS.TITLE
RELIGIONS	COUNTRY	RELIGIONS.COUNTRY
	RELIGION	RELIGIONS.RELIGION
	PERCENT	RELIGIONS.PERCENT

Full column name format is tablename.columnname

Joins

continued

Columns from two or more tables can be combined into a single result table using a technique called joining. The principles behind joining are illustrated in the three steps below. Starting with two tables that have at least one column in common...

PERSONS

PERSON	NAME	JOB
1	Einstein	S
2	Dickens	W
3	Dickinson	W

JOBS

JOB	TITLE
S	Scientist
W	Writer
E	Entertainer

1) We append ALL the rows of the second table onto EACH of the rows of the first...

PERSON	NAME	JOB	JOB	TITLE
1	Einstein	S	S	Scientist
1	Einstein	S	W	Writer
1	Einstein	S	E	Entertainer
2	Dickens	W	S	Scientist
2	Dickens	W	W	Writer
...	...	...	...	...

2) Then we select only the rows where the common columns contain equal values...

PERSON	NAME	JOB	JOB	TITLE
1	Einstein	S	S	Scientist
2	Dickens	W	W	Writer
3	Dickinson	W	W	Writer

3) Finally, we eliminate the redundant column. The result is:

PERSON	NAME	JOB	TITLE
1	Einstein	S	Scientist
2	Dickens	W	Writer
3	Dickinson	W	Writer

Joins

continued

To join two or more tables using SQL, 1) list all the required tables in the FROM clause; 2) enter the appropriate comparison(s) in the WHERE clause; and 3) specify which columns should appear in the result in the SELECT clause. Like so:

PERSONS

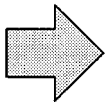
PERSON	NAME	JOB
1	Einstein	S
2	Dickens	W
3	Dickinson	W

JOBS

JOB	TITLE
S	Scientist
W	Writer
E	Entertainer

SQL STATEMENT

```
SELECT PERSON, NAME, PERSONS.JOB, TITLE
FROM PERSONS, JOBS
WHERE PERSONS.JOB = JOBS.JOB
```



PERSON	NAME	JOB	TITLE
1	Einstein	S	Scientist
2	Dickens	W	Writer
3	Dickinson	W	Writer

Multiple tables are specified in the FROM clause

Join comparisons are entered in the WHERE clause

Full column names are employed as necessary

Joins

continued

Note that the SELECT clause now allows full column names, FROM supports a table list, and the WHERE clause includes column-to-column comparisons.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table list	PERSONS, JOBS
WHERE	comparisons	PERSONS.JOB = JOBS.JOB
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC
	func [DESC] list	COUNT (*) DESC

Exercises

1. List the name and native language of each scientist born in the 20th century. There are 6 rows in the result.

2. List the country, the GNP, and the military budget for all countries with over 100 million people. There are 8 rows in the result.

3. Revise the previous query to include the percent of GNP spent on the military.

Extra Credit

4. Approximately how many members of the German military are Protestant?

Table Aliases

The keyword AS can be used in the FROM clause to define a table alias — a user-assigned name for a table. Table aliases can be specified for any table, but they are most frequently used in multi-table queries to shorten full column names.

SQL STATEMENT

```
SELECT PERSON, NAME, PERSONS.JOB, TITLE
FROM PERSONS, JOBS
WHERE PERSONS.JOB = JOBS.JOB
```

SQL STATEMENT

```
SELECT PERSON, NAME, P.JOB, TITLE
FROM PERSONS AS P, JOBS AS J
WHERE P.JOB = J.JOB
```

Table aliases are defined in the FROM clause

Table aliases can be used in any clause

Once an alias is defined, the original cannot be used

AS is optional in some dialects

Table Aliases

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of table aliases in the FROM clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS, JOBS AS J
WHERE	comparisons	PERSONS.JOB = J.JOB
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC
	func [DESC] list	COUNT (*) DESC

Exercises

1. Using table aliases, display country, population, and number of troops for each country with more than 100 million people. The are 8 rows in the result.

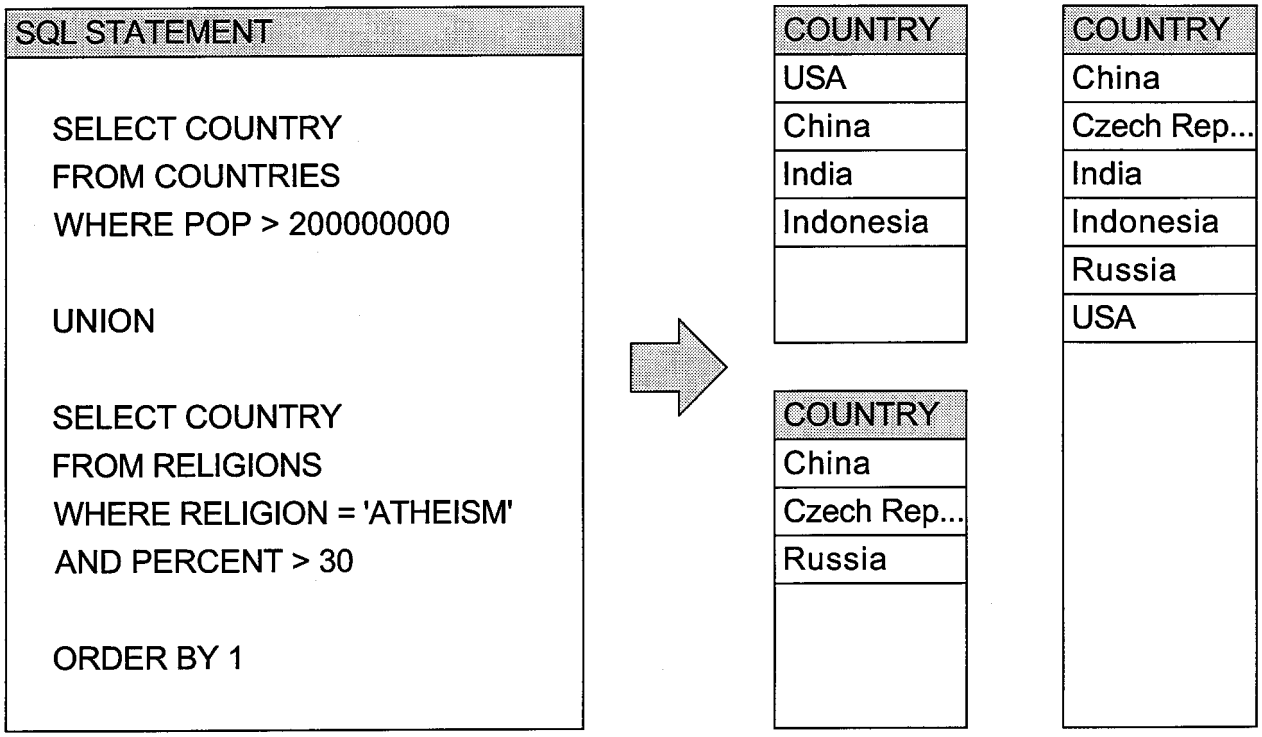
2. List all languages spoken in countries that are over 90 percent Muslim. Use table aliases, and eliminate duplicates from the result. There are 4 such languages.

Extra Credit

3. List the name, job title, and native language for all persons in the database born before January 1, 1400. There are 10 such people.

Unions

A union combines the rows of two similar queries into a single result with no duplicates. The keyword UNION is placed between the queries. The ORDER BY clause, if used, must specify column positions and appear in the final query only.



The number and type of columns must match

The UNION keyword is placed between the queries

ORDER BY, if used, must be the last clause

ORDER BY must specify column positions

Duplicate rows are not included in the result

Unions

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of the UNION clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS, JOBS AS J
WHERE	comparisons	PERSONS.JOB = J.JOB
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
UNION	query	SELECT ...
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC
	func [DESC] list	COUNT (*) DESC

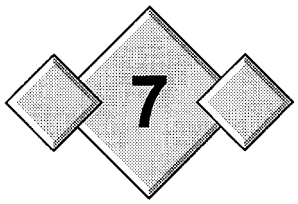
Exercises

1. List the eight countries that are more than 40% Protestant; then list the four that speak German. Union the previous two. How many rows are in the final result?

2. Union all countries that have produced a scientist, with all countries that have a military budget over 10 billion. Sort the result by country name. There are 12 rows.

Extra Credit

3. List person, country, and language names where the second letter is Z.



QUERIES WITHIN QUERIES

Single-Valued Subqueries

Multi-Valued Subqueries

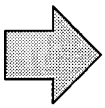
Correlated Subqueries

Single-Valued Subqueries

A single-valued subquery is a query that 1) produces a result with a single column and a single row; and 2) is 'nested' in the WHERE clause of another query. Subqueries must be enclosed in parentheses.

SQL STATEMENT

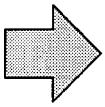
```
SELECT AVG ( POP )
FROM COUNTRIES
```



AVG(POP)
30150531

SQL STATEMENT

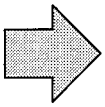
```
SELECT COUNTRY, POP
FROM COUNTRIES
WHERE POP > 30150531
```



COUNTRY	POP
Germany	81337541
USA	263814032
England	58295119
Argentina	34292742
Colombia	36200251
Mexico	93985848

SQL STATEMENT

```
SELECT COUNTRY, POP
FROM COUNTRIES
WHERE POP >  ( SELECT AVG ( POP )
               FROM COUNTRIES )
```



COUNTRY	POP
Germany	81337541
USA	263814032
England	58295119
Argentina	34292742
Colombia	36200251
Mexico	93985848

Single-valued queries can be used in comparisons

Subqueries are enclosed in parentheses

Single-Valued Subqueries

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of single-valued subqueries in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS, JOBS AS J
WHERE	comparisons	PERSONS.JOB = J.JOB
	column oper (subquery)	POP > (SELECT AVG (POP) ...)
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
UNION	query	SELECT ...
ORDER BY	sort list	LANGUAGE DESC, COUNTRY

Exercises

1. Find the largest population of all countries. Now use that query as a subquery to display all columns for the country with the largest population.

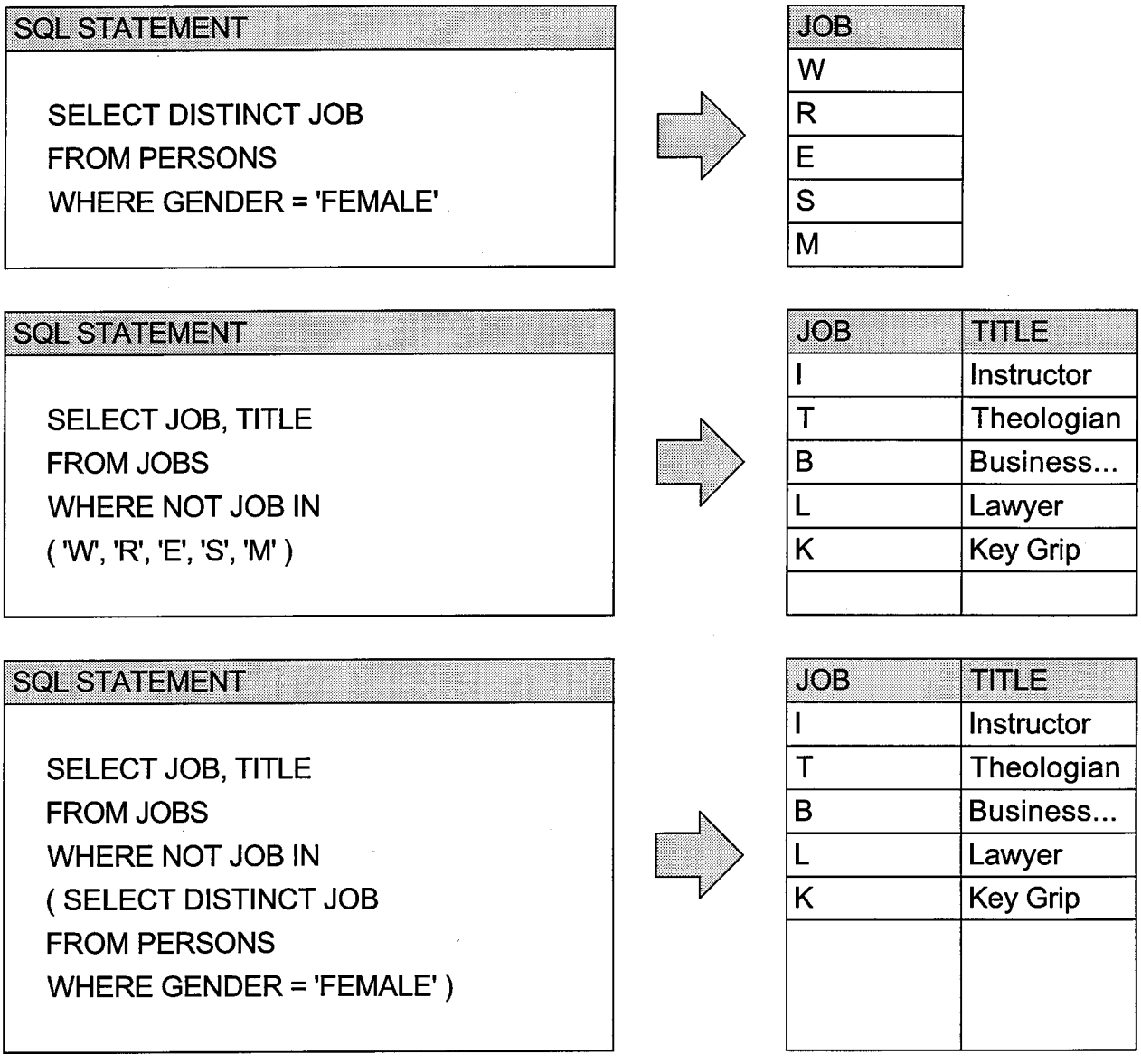
2. Display all columns for armies whose budget is higher than the average budget of all armies. There are 5 rows in the result table.

Extra Credit

3. Show the native language, name, job title, and birth date of all people with the same job as Luther. Sort by name within language. The result contains 13 rows.

Multi-Valued Subqueries

A multi-valued subquery is a query that 1) returns a single-column result with zero, one, or more rows; and 2) is 'nested' in the WHERE clause of another query. Multi-valued subqueries always follow the IN operator.



Multi-valued subqueries follow the IN operator

Multi-Valued Subqueries

continued

The current syntax of a SQL SELECT statement appears below. Note the addition of multi-valued subqueries in the WHERE clause.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS, JOBS AS J
WHERE	comparisons	PERSONS.JOB = J.JOB
	column oper (subquery)	POP > (SELECT AVG (POP) ...)
	column IN (subquery)	JOB IN (SELECT DISTINCT JOB ...)
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
UNION	query	SELECT ...
ORDER BY	sort list	LANGUAGE DESC, COUNTRY

Exercises

1. Make a list of countries whose literacy rate is less than 50%. Now, use that query to show all columns for writers born in any of those countries. There are 2 rows.

2. Display all columns for people who were born in countries that are over 95% Catholic. There are 13 such people.

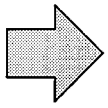
Extra Credit

3. Show all columns for military leaders born in English-speaking countries that have less than the average number of troops in their army. There are 2 rows.

Correlated Subqueries

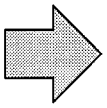
A correlated subquery is a single- or multi-valued subquery that references the outer query via a table alias defined in the outer query's FROM clause. The subquery is executed once for each row of the outer query.

SQL STATEMENT
SELECT JOB, NAME FROM PERSONS WHERE JOB = 'S' AND BDATE = (SELECT MIN (BDATE) FROM PERSONS WHERE JOB = 'S')



JOB	NAME
S	Magnus

SQL STATEMENT
SELECT JOB, NAME FROM PERSONS AS P WHERE BDATE = (SELECT MIN (BDATE) FROM PERSONS WHERE JOB = P.JOB)



JOB	NAME
E	Hepburn
W	Shikabu
M	Montcalm
B	Dupont
S	Magnus
T	Paul
R	Jean

Correlated subqueries are executed multiple times

Correlated subqueries require table aliases

Correlated Subqueries

continued

The current syntax of a SQL SELECT statement appears below. Note that correlated subqueries can be single-valued or multi-valued, and require table aliases.

SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
SELECT [DISTINCT]	*	*
	col [AS alias] list	NAME, PERSONS.JOB, TITLE
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS AS P
WHERE	comparisons	PERSONS.JOB = J.JOB
	column oper (subquery)	BDATE = (SEL ... WHERE JOB = P.JOB)
	column IN (subquery)	BDATE IN (SEL ... WHERE JOB = P.JOB)
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
UNION	query	SELECT ...
ORDER BY	sort list	LANGUAGE DESC, COUNTRY

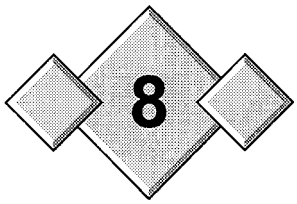
Exercises

1. Show the job and name for the youngest person from each profession. The result has 7 rows and includes such people as Grisham, Mountbatten, and Forbes.

2. Using a single query with a correlated subquery, show all columns for the youngest male and the youngest female. What else do they have in common?

Extra Credit

3. Find the dominant religion in each German-speaking country. The result table contains 4 rows.



MAINTAINING TABLES

Inserting Rows

Updating Rows

Deleting Rows

Transactions

Inserting Rows

New rows may be inserted into a table one at time using the SQL INSERT statement. A table name and column list are specified in the INTO clause, and a value list, enclosed in parentheses, is entered in the VALUES clause.

COUNTRIES (BEFORE)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Yugoslavia	11101833	39449	10000	Serbo-Cro...	89

SQL STATEMENT

```
INSERT
INTO COUNTRIES ( COUNTRY, LITERACY, AREA, POP )
VALUES ( 'Beulah', 100, 1146807, 144000 )
```

COUNTRIES (AFTER)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Yugoslavia	11101833	39449	10000	Serbo-Cro...	89
Beulah	144000	1146807	-	-	100

A table name is specified in the INTO clause

A column list follows the table name

Appropriate values are listed in the VALUES clause

Unlisted columns are assigned default values

Inserting Rows

continued

The general syntax of a SQL INSERT statement appears below. It contains three clauses: INSERT, INTO, and VALUES.

INSERT SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
INSERT		
INTO	table (col list)	COUNTRIES (COUNTRY, POP)
VALUES	(value list)	('Beulah', 144000)

Exercises

1. Select all columns and all rows of the JOBS table. Now add a row to the JOBS table using 'A' for the job and 'Author' for the title.

2. Select all columns for persons with a person number greater than or equal to 500. Now add yourself to the PERSONS table as person number 500. Fill in all of the columns. If your job is not in the JOBS table, use job 'R'.

3. Add a friend to the PERSONS table. Use 501 as the person number.

Updating Rows

You can modify one or more rows of a table using the UPDATE statement. The table is specified in the UPDATE clause, and new values are entered in the SET clause. All rows are updated unless the WHERE clause is used to restrict the operation.

COUNTRIES (BEFORE)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Yugoslavia	11101833	39449	10000	Serbo-Cro...	89
Beulah	144000	1146807	-	-	100

SQL STATEMENT
UPDATE COUNTRIES SET AREA = 53501998, LANGUAGE = 'Hebrew', POP = 100000000 WHERE COUNTRY = 'Beulah'

COUNTRIES (AFTER)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Yugoslavia	11101833	39449	10000	Serbo-Cro...	89
Beulah	100000000	53501998	-	Hebrew	100

A table name is specified in the UPDATE clause

New values are specified in the SET clause

The WHERE clause specifies which rows to update

Omitting the WHERE clause updates all rows

Updating Rows

continued

The general syntax of a SQL UPDATE statement appears below. It contains three clauses: UPDATE, SET, and the optional WHERE clause.

UPDATE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
UPDATE	table	COUNTRIES
SET	new value list	AREA = 7992, LANGUAGE = 'Hebrew'
WHERE	comparisons	COUNTRY = 'Beulah'

Exercises

1. Change all of the writers to authors. Run a query to verify your changes.

2. Change your birth date to a more recent date, and your job to entertainer. But don't kid yourself — you're not getting any younger. Run a query to verify your changes.

3. Now change the authors back to writers. Verify the result.

Deleting Rows

You can delete one or more rows from a table using the DELETE statement. A table is specified in the FROM clause, and optional comparisons are entered in the WHERE clause. Omitting the WHERE clause deletes all rows.

COUNTRIES (BEFORE)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Yugoslavia	11101833	39449	10000	Serbo-Cro...	89
Beulah	100000000	53501998	-	Hebrew	100

SQL STATEMENT

```
DELETE
FROM COUNTRIES
WHERE COUNTRY = 'Yugoslavia'
```

COUNTRIES (AFTER)

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Algeria	28539321	919595	89000	Arabic	57
Beulah	100000000	53501998	-	Hebrew	100

A table name is specified in the FROM clause

The WHERE clause specifies which rows to delete

Omitting the WHERE clause deletes all rows

Deleting Rows

continued

The general syntax of a SQL DELETE statement appears below. It contains three clauses: DELETE, FROM, and the optional WHERE clause.

DELETE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
DELETE		
FROM	table	COUNTRIES
WHERE	comparisons	COUNTRY = 'Yugoslavia'

Exercises

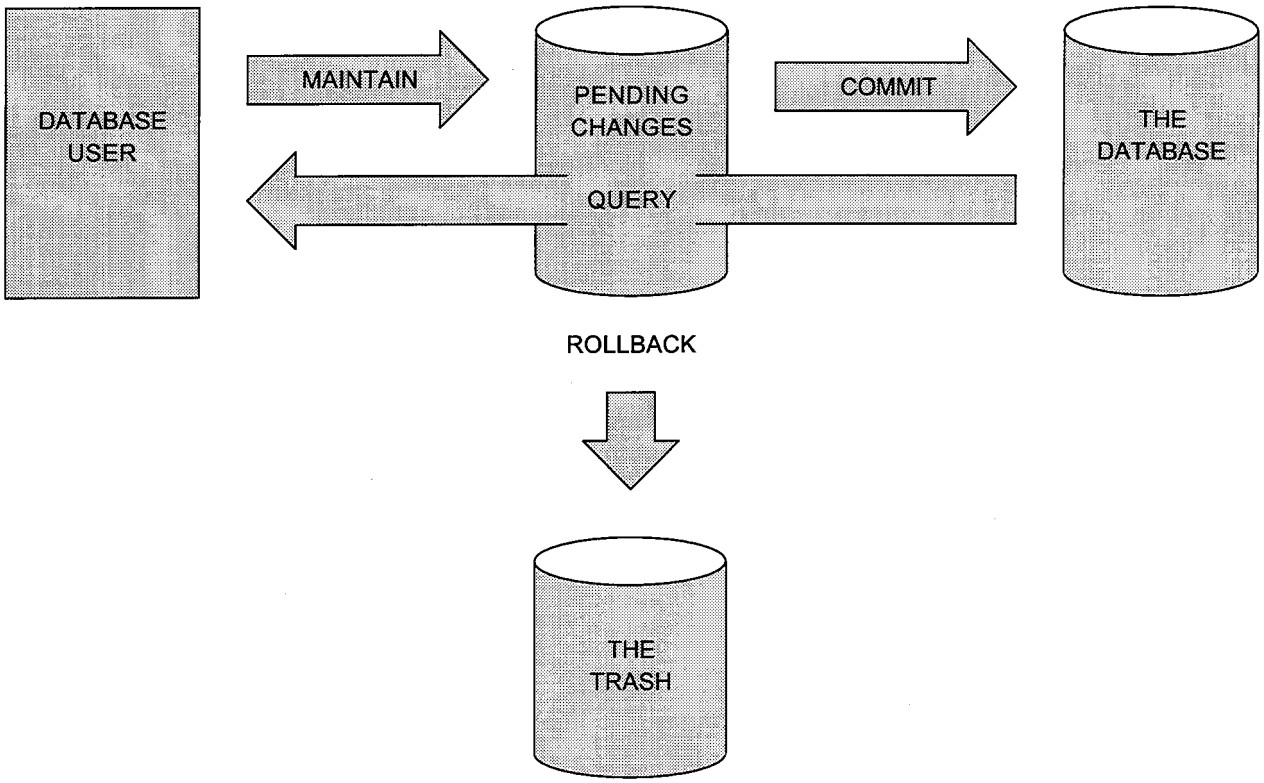
1. Delete your friend and yourself from the database using separate statements. Run a query to verify that you no longer exist. Pinch yourself — you really do.

2. Run a query to count the number of people born in the 18th century. There are 27 such people. Now delete them. Re-run the count to verify the changes.

3. How many rows are there in the RELIGIONS table? Delete all of the rows where the religion includes 'orthodox' in the name. There should be 369 rows left.

Transactions

A transaction is a collection of related maintenance operations whose effects can be permanently applied to, or completely removed from a database. Transactions can include insert, update, and/or delete operations against one or more tables.



ROLLBACK permanently reverses transactions

The ROLLBACK statement 'undoes' all maintenance operations performed since the beginning of the session, or since the previous COMMIT, whichever is more recent.

COMMIT permanently applies transactions

The COMMIT statement permanently applies all maintenance operations performed since the beginning of the session, or since the previous COMMIT to the database.

Transactions

continued

The general syntax of SQL transaction statements appears below. There are two distinct statements, ROLLBACK and COMMIT.

ROLLBACK SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
ROLLBACK		

COMMIT SYNTAX SUMMARY

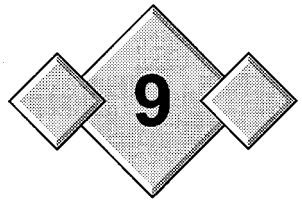
CLAUSE	PARAMETERS	EXAMPLE
COMMIT		

Exercises

1. Rollback all of the changes made to the database so far. Is 'A' in the JOBS table? How many rows are now in the RELIGIONS table? There should be 389.

2. Delete all rows from PERSONS and ARMIES. See if they are gone. Rollback this transaction. See if they are back.

3. Add yourself back into the PERSONS table. Commit yourself. Rollback. Are you still there? If you are, delete yourself and commit again.



DEFINING DATABASE OBJECTS

Defining Tables

Loading Tables

Integrity Constraints

Defining Indices

Defining Views

Defining Tables

New tables are defined in a database by entering a CREATE TABLE statement. In its simplest form, the CREATE TABLE statement includes a new table name and one or more column definitions. Tables are removed using the DROP TABLE statement.

SQL STATEMENT

```
CREATE TABLE WRITERS
(
NAME CHAR ( 20 ),
BDATE DATE,
GENDER CHAR ( 6 ),
COUNTRY CHAR ( 20 ),
SALES NUMERIC ( 9, 2 )
)
```

SQL STATEMENT

```
DROP TABLE WRITERS
```

Tables are added with the CREATE TABLE statement

One CREATE TABLE statement is required for each table. In most SQL dialects, table names cannot exceed 18 characters in length, must begin with a letter, and cannot contain any characters other than letters, digits, and underscores.

Columns are defined in parentheses

A column definition includes a name, data type, and optional size (in parentheses). Column definitions are separated by commas. Data types differ between systems.

Tables are removed with the DROP TABLE statement

The DROP TABLE statement permanently removes a specified table from the database. Both CREATEs and DROPs are automatically committed upon execution.

Defining Tables

continued

The general syntax of a SQL CREATE TABLE statement appears below. The syntax of a SQL DROP TABLE statement is also shown.

CREATE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
CREATE	TABLE table	TABLE WRITERS
(		
	col type (size) list	NAME CHAR (20), BDATE DATE, SALES NUMERIC (9, 2)
)		

DROP SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
DROP	TABLE table	TABLE WRITERS

Exercises

1. Create a table called SCIENTISTS with four columns: NAME, BDATE, GENDER, and COUNTRY. Make text columns capable of holding 20 characters.

2. Add a few rows to the new table. Verify that the table and rows exist.

3. Drop the SCIENTISTS table. Run a query to verify that the table no longer exists.

Loading Tables

A batch of rows can be added to a table using the SQL INSERT statement. The VALUES clause (which supports one row at a time) is omitted, and a query that returns multiple rows is entered in its place. The query is enclosed in parentheses.

WRITERS (BEFORE)

NAME	BDATE	GENDER	COUNTRY	SALES

SQL STATEMENT

```
INSERT
INTO WRITERS ( NAME, BDATE, GENDER, COUNTRY )
( SELECT NAME, BDATE, GENDER, COUNTRY
FROM PERSONS
WHERE JOB = 'W' )
```

WRITERS (AFTER)

NAME	BDATE	GENDER	COUNTRY	SALES
Dickens	1812-07-22	Male	England	-
Dickinson	1830-07-15	Female	USA	-
Holmes	1809-04-09	Male	USA	-
Moliere	1622-07-03	Male	France	-

Multiple rows can be added to a table using INSERT

A query is specified instead of the VALUES clause

The query is enclosed in parentheses

Loading Tables

continued

The revised syntax of a SQL INSERT statement appears below. Note the addition of the query parameter, which is mutually exclusive with the VALUES clause.

INSERT SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
INSERT		
INTO	table (col list)	COUNTRIES (COUNTRY, POP)
VALUES	(value list)	('Beulah', 144000)
	(query)	(SELECT ...)

Exercises

1. Recreate the SCIENTISTS table with four columns: NAME, BDATE, GENDER, and COUNTRY. Make text columns capable of holding 20 characters.

2. Load the SCIENTISTS table with only female scientists. How many are there?

3. Load all of the male scientists into the SCIENTISTS table. How many now?

4. Insert all of the theologians into the SCIENTISTS table, since systematic theology is the greatest of all sciences. How many rows are in the table now?

5. Drop the SCIENTISTS table.

Integrity Constraints

The values that appear in a column can be restricted by specifying constraints in the CREATE statement. Constraints are enforced by the database at insert, update, and delete time. The two simplest, NOT NULL and DEFAULT, are entered as follows:

SQL STATEMENT

```
CREATE TABLE WRITERS
(
NAME CHAR ( 20 ) NOT NULL,
BDATE DATE,
GENDER CHAR ( 6 ) DEFAULT 'Male',
COUNTRY CHAR ( 20 ),
SALES NUMERIC ( 9, 2 ) DEFAULT 0
)
```

NOT NULL prevents null values

Since null values are generally considered undesirable, the NOT NULL constraint is provided to prevent null values from being stored in a column. NOT NULL is specified after the data type and size for a given column, as shown above.

DEFAULT provides initial values for new rows

Entering the keyword DEFAULT, followed by a value (as shown above), tells the database to assign that value in the specified column whenever a row is added to that table and no user-specified value is provided.

Integrity Constraints

continued

The UNIQUE and CHECK constraints provide a higher level of integrity by forcing values in a table to meet additional conditions. Both UNIQUE and CHECK may apply to one column, or to a combination of columns in a given table.

SQL STATEMENT

```
CREATE TABLE WRITERS
(
NAME CHAR ( 20 ) NOT NULL,
BDATE DATE,
GENDER CHAR ( 6 ) DEFAULT 'Male',
COUNTRY CHAR ( 20 ),
SALES NUMERIC ( 9, 2 ) DEFAULT 0,
UNIQUE ( NAME ),
CHECK (GENDER IN ( 'Male', 'Female' )),
CHECK (SALES >= 0)
)
```

UNIQUE prevents duplicate values

The keyword UNIQUE, followed by a list of columns (in parentheses), prevents duplicate values from being stored in the specified combination of columns. UNIQUE constraints are entered below column definitions.

CHECK insures valid values

The keyword CHECK, followed by a comparison (in parentheses), prevents rows that do not satisfy the CHECK criteria from being stored in the table. CHECK constraints are entered below column definitions.

Integrity Constraints

continued

A primary key is the column (or collection of columns) that is always unique for each row in a table. A foreign key is a column whose values refer to a primary key in some other table. PRIMARY KEY and FOREIGN KEY constraints can be defined like so:

SQL STATEMENT

```
CREATE TABLE WRITERS
(
NAME CHAR ( 20 ),
BDATE DATE,
GENDER CHAR ( 6 ) DEFAULT 'Male',
COUNTRY CHAR ( 20 ),
SALES NUMERIC ( 9, 2 ) DEFAULT 0,
CHECK ( GENDER IN ( 'Male', 'Female' ) ),
CHECK ( SALES >= 0 ),
PRIMARY KEY ( NAME ),
FOREIGN KEY ( COUNTRY )
REFERENCES COUNTRIES ( COUNTRY )
)
```

PRIMARY KEY insures unique rows

Entering PRIMARY KEY, followed by a column list in parentheses, tells the database to prevent nulls and duplicates for the specified combination of columns.

FOREIGN KEY references primary key values

Foreign keys are defined by entering FOREIGN KEY, a column list, REFERENCES, the referenced table, and that table's primary key (as shown above). The system insures that values in foreign key columns exist in the referenced table.

Integrity Constraints

continued

The revised syntax of a CREATE TABLE statement appears below. Note the NOT NULL, DEFAULT, UNIQUE, CHECK, PRIMARY KEY and FOREIGN KEY constraints.

CREATE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
CREATE	TABLE table	TABLE WRITERS
(		
	col type (size) list	NAME CHAR (20), BDATE DATE, SALES NUMERIC (9, 2)
	NOT NULL	NOT NULL
	DEFAULT value,	DEFAULT 0,
UNIQUE	(col list),	(NAME),
CHECK	(comparison),	(GENDER IN ('Male', 'Female')),
PRIMARY KEY	(col list),	(NAME),
FOREIGN KEY	(col list) REFERENCES table (col list)	(COUNTRY) REFERENCES COUNTRIES (COUNTRY)
)		

Exercises

1. Create a THEOLOGIANS table with these four columns: NAME, BDATE, GENDER, and COUNTRY columns. Make text columns hold 20 characters each. Gender cannot be null, defaults to 'male', and must be either 'male' or 'female'; NAME is the primary key; and COUNTRY is a foreign key that references COUNTRIES. Load the table with the appropriate rows from the PERSONS table.

2. Try adding a row with an invalid country, like Prussia. Now try adding a row with a null gender (nulls are entered as NULL, and are not enclosed in quotes).

3. Drop the THEOLOGIANS table.

Defining Indices

An index is an invisible data structure that makes a table quicker to access. Indices are created with the CREATE INDEX statement, are automatically used by SQL when appropriate, and are removed from the database via the DROP INDEX statement.

SQL STATEMENT
CREATE INDEX GJX ON PERSONS (GENDER, JOB)

SQL STATEMENT
DROP INDEX GJX

Indices are added with the CREATE INDEX statement

Indices are defined using the CREATE INDEX statement. Index names must be unique within a database, and are used internally by the system. Index names are also used when removing an existing index.

Indices are used automatically

Queries in SQL take less time to execute when appropriate indices are defined. Indices, however, consume additional disk space, and slow down maintenance operations (since the indices must be updated as well).

Indices are removed with the DROP INDEX statement

To remove an index from a database, use the DROP INDEX statement. The table's data remains; disk space previously occupied by the index is freed for other purposes. Both CREATEs and DROPs are automatically committed upon execution.

Defining Indices

continued

The general syntax of a SQL CREATE INDEX statement appears below. The syntax of a SQL DROP INDEX statement is also shown.

CREATE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
CREATE	INDEX index	INDEX GJX
ON	table (col list)	PERSONS (GENDER, JOB)

DROP SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
DROP	INDEX index	INDEX GJX

Exercises

1. Create a table called REGIONS with one CHAR(10) column called NAME. Insert four rows, setting the values to North, South, East, and West respectively. Now insert the table into itself until you have about 65,536 rows.

2. Count the rows, subtotaled by region name. Note the time required to execute the query.

3. Create an index called RX on the NAME column of the REGIONS table. Rerun the query from the last exercise. Is it faster, slower, or about the same? Why?

4. Drop the index and the table.

Defining Views

A view is a virtual table that is derived from other tables (or views) in a database. Views are not physically stored in the database, but can be queried as if they were actual tables. The following examples show how to create and remove views:

SQL STATEMENT
CREATE VIEW PTV (PERSON, NAME, TITLE) AS SELECT PERSON, NAME, TITLE FROM PERSONS AS P, JOBS AS J WHERE P.JOB=J.JOB

SQL STATEMENT
DROP VIEW PTV

Views are added with the CREATE VIEW statement

Views are handy for frequently accessed information that is difficult to derive, such as summaries involving multiple tables. Any query statement can be used to define a view, with the exception that view definitions cannot contain ORDER BY clauses.

Views may be updateable

Typically, a view created from a single table with all non-null columns included can be updated; other views cannot. Information updated via a view is changed in the underlying tables involved in that view, as appropriate.

Views are removed with the DROP VIEW statement

To remove a view from a database, use the DROP VIEW statement. Note that removing a view from a database does not remove data from any of the underlying tables. Both CREATEs and DROPs are automatically committed upon execution.

Defining Views

continued

The general syntax of a SQL CREATE VIEW statement appears below. The syntax of a SQL DROP VIEW statement is also shown.

CREATE SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
CREATE	VIEW view (col list)	VIEW PTV (PERSON, NAME, TITLE)
AS	query	SELECT ...

DROP SYNTAX SUMMARY

CLAUSE	PARAMETERS	EXAMPLE
DROP	VIEW view	VIEW PTV

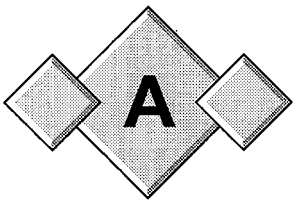
Exercises

1. Create a view called IV that contains the same columns as the PERSONS table, but only those persons who were born in Israel. Query the view to verify it.

2. Insert a friend into the IV view, using 502 as the person number and Israel as the country. Now insert a long-lost friend into IV, using 503 as the person number and setting the country to USA. What happened to your long-lost friend?

3. Create a view called IJV, based on the IV view. Include the NAME, BDATE, and COUNTRY columns, together with the TITLE column from the JOBS table.

4. Drop all of your views. Thank you. Bye bye.



APPENDICES

The Exercise Database

Answers to Exercises

Syntax Summary

A Brief Critique

Recommended Reading

Index

The Exercise Database

The following tables constitute the logical database for all examples and exercises. Note that the actual tables installed in your classroom will have more rows.

PERSONS

PERSON	NAME	BDATE	GENDER	COUNTRY	JOB
1	Einstein	1879-03-20	Male	Germany	S
2	Dickens	1812-07-22	Male	England	W
3	Dickinson	1830-07-15	Female	USA	W

COUNTRIES

COUNTRY	POP	AREA	GNP	LANGUAGE	LITERACY
Germany	81337541	137823	1331000	German	100
USA	263814032	3679192	6380000	English	95
England	58295119	94251	980200	English	99

ARMIES

COUNTRY	BUDGET	TROOPS	TANKS	SHIPS	PLANES
Germany	35200	367300	2855	32	300
USA	280600	1650500	14524	230	10600
England	41700	254300	921	52	340

JOBS

JOB	TITLE
S	Scientist
E	Entertainer
W	Writer

RELIGIONS

COUNTRY	RELIGION	PERCENT
Germany	Protestant	45
Germany	Catholic	37
England	Catholic	30

POP is in people. AREA is in square miles. GNP is in millions of dollars. LANGUAGE is the principal language of the country. LITERACY is in percent. BUDGET is in millions of dollars. TROOPS are active troops only. These tables are for illustration purposes only. No animals were harmed in the production of this database.

Answers to Exercises

Page 10

1. O Most Kind and Gracious Instructor, could you please give us a brief tutorial in the use of the classroom database?

2. select *
   from persons

3. select *
   from countries

4. select *
   from errors

   This query generates a syntax error.

5. select *
   from armies

6. select *
   from syscolumns

7. select *
   from how-would-we-know-what-you-picked

Page 12

1. select job
   from jobs

2. select name,bdate
   from persons

3. select percent,religion,country
   from religions

4. select country,pop,area,gnp,language,
   literacy,country
   from countries

   select *,country
   from countries

Answers to Exercises

continued

Page 16

1. select *
   from countries

   select *
   from countries
   where area<30

2. select name,country
   from persons

   select name,country
   from persons
   where country='canada'

3. select name,bdate
   from persons
   where bdate>'1964-01-01'

4. select bdate
   from persons
   where name='o''toole'

   Note the consecutive quotes in the string.
   O'Toole's birth date is 1932-08-12.

Page 19

1. select *
   from persons
   where country='ireland'

   select *
   from persons
   where country='ireland'
   order by name

2. select *
   from countries
   where language='german'

   select *
   from countries
   where language='german'
   order by gnp desc

3. select country,job,name
   from persons
   where country='italy'
   order by job,name

4. select *
   from armies
   order by budget desc

   The USA has the largest budget.

   select *
   from armies
   order by budget

   Vietnam has the smallest budget.

   China has the most troops.

   Russia has the most tanks.

   Russia has the most ships.

   The USA has the most planes.

Answers to Exercises

continued

Page 21

1. select job
   from persons

   select distinct job
   from persons

2. select language
   from countries
   where literacy<30
   order by language

   select distinct language
   from countries
   where literacy<30
   order by language

3. select country
   from persons
   where job='s'
   order by country

   select distinct country
   from persons
   where job='s'
   order by country

4. select religion
   from religions
   where percent<5
   order by religion

   select distinct religion
   from religions
   where percent<5
   order by religion

Page 24

1. select country
   from countries
   where country like '%guinea%'

2. select *
   from persons
   where name like '_z%'

3. select *
   from persons
   where bdate like '%07-15'

4. select *
   from persons
   where name like '%''%'

5. select distinct religion
   from religions
   where religion like '%orthodox%'
   order by religion

Answers to Exercises

continued

Page 26

1. select *
   from persons
   where job='s'
   and country='germany'

2. select *
   from countries
   where gnp<3000
   and literacy<40
   order by gnp

3. select country,literacy
   from countries
   where literacy>=55
   and literacy<=60

4. select *
   from persons
   where country='england'
   and bdate like '15%'
   order by name

5. select *
   from armies
   where troops>100000
   and tanks<1000
   and ships<1000
   and planes<1000

Page 28

1. select *
   from countries
   where area between 5 and 75

2. select *
   from countries
   where pop between 100000 and 200000

3. select *
   from armies
   where planes between 301 and 399

4. select *
   from armies
   where troops>=300000
   and budget between 10000 and 100000

Answers to Exercises

continued

Page 30

1. select *
   from armies
   where country='israel'
   or country='iraq'

2. select *
   from persons
   where bdate='1835-10-19'
   or bdate='1917-02-06'

3. select name,bdate
   from persons
   where name='Poe'
   or name='Hugo'
   or name='Dahl'

4. select *
   from persons
   where bdate like '12%'
   or bdate like '14%'

   Five people were born during this period.

Page 32

1. select *
   from persons
   where name in ('einstein','galilei','newton')

2. select *
   from countries
   where literacy in (20,40,60)

3. select name,country
   from persons
   where job = 's'
   and country in ('germany','austria','italy')

4. select *
   from persons
   where job in ('w','e','b') and
   country in ('germany','austria','italy')

Answers to Exercises

continued

Page 34

1. select *
   from countries
   where gnp is null

2. select *
   from persons
   where job='e'
   and gender is null

3. select *
   from persons
   where country='israel'
   and bdate is null

4. select *
   from countries
   where area<10
   and gnp>250

   There is one such country.

   select *
   from countries
   where area<10
   and gnp<=250

   There are two such countries.

   select *
   from countries
   where area<10

   You would guess the answer to be three, the total number of rows returned by the first two queries. But there are actually four. One country, because it has a null gnp, does not appear in either of the first two queries.

Page 36

1. select *
   from persons
   where country='germany'
   and (job='b' or job='t')
   order by name

2. select *
   from persons
   where country='germany'
   and not (job='b' or job='t')
   order by name

3. select *
   from armies
   where not country in ('usa', 'russia')
   and budget>30000

   ...3 rows in the result.

Answers to Exercises

continued

Page 40

1. select pop*.2
   from countries
   where country='canada'

   5,686,909 Canadian-Americans, eh?

2. select budget*1000000/troops
   from armies
   where country='usa'

   $170,000.09 is spent per trooper in the USA.

   select budget*1000000/troops
   from armies
   where country='China'

   $921.50 is spent per trooper in China.

3. select tanks+ships+planes
   from armies
   where country='usa'

   The total is 25,354 military vehicles.

4. select budget*1000000*.45/125
   from armies
   where country='usa'

   1,010,160,000 hammers!

Page 42

1. select country,pop,area,pop/area
   from countries
   where pop/area<7
   order by pop/area desc

2. select country,pop,literacy,literacy/100*pop
   from countries
   where literacy/100*pop>100000000

3. select country,budget*1000000*.3/650/troops
   from armies
   where budget*1000000*.3/650/troops>10
   order by budget*1000000*.3/650/troops desc

Answers to Exercises

continued

Page 44

1. select country,pop,literacy,
   literacy/100*pop as readers
   from countries
   where readers>100000000

2. select country,pop,gnp,
   gnp*1000000/pop as gpp
   from countries
   where gpp>20000
   order by gpp desc

3. select country,
   troops*.20*20000 as fission_fund
   from armies
   where fission_fund>=2000000000
   order by fission_fund desc

   There are 11 contributing countries.

Page 48

1. select sum(troops)
   from armies

2. select
   min(literacy),max(literacy),avg(literacy)
   from countries
   where language='french'

3. select
   count(country),count(distinct language)
   from countries

Answers to Exercises

continued

Page 50

1. select language,sum(pop)
   from countries
   where language in
   ('hebrew','spanish','english','french')
   group by language

2. select language,min(literacy),
   max(literacy),avg(literacy)
   from countries
   where language in
   ('hebrew','spanish','english','french')
   group by language

3. select country,gender,count(person)
   from persons
   where country in
   ('canada','france')
   group by country,gender
   order by country,gender

4. select job,gender,count(person)
   from persons
   where not gender is null
   group by job,gender
   order by job,gender

Page 53

1. select country,count(person)
   from persons
   where job='w'
   group by country
   order by count(person)

2. select language,sum(pop)
   from countries
   group by language
   having sum(pop)<1000000

3. select language,min(literacy),
   max(literacy),avg(literacy)
   from countries
   group by language
   having min(literacy)<>max(literacy)

4. select language,sum(area),sum(pop)
   from countries
   group by language
   having count(*)>1
   order by sum(area) desc

Answers to Exercises

continued

Page 58

1. select name,language
   from persons,countries
   where persons.country=countries.country
   and job='s'
   and bdate like '19%'

2. select countries.country,gnp,budget
   from countries,armies
   where countries.country=armies.country
   and pop>100000000

3. select countries.country,gnp,budget,
   budget/gnp*100
   from countries,armies
   where countries.country=armies.country
   and pop>100000000

4. select troops*percent/100
   from armies,religions
   where armies.country=religions.country
   and armies.country='germany'
   and religion='protestant'

   Assuming troops in the German military are a cross-section of the population as a whole, about 165,285 of them are Protestant.

Page 60

1. select c.country,pop,troops
   from countries as c,armies as a
   where c.country=a.country
   and pop>100000000

2. select distinct language
   from country as c,religion as r
   where c.country=r.country
   and religion='muslim'
   and percent>90

3. select name,title,language
   from persons as p,jobs as j,countries as c
   where p.job=j.job
   and p.country=c.country
   and bdate<'1400-01-01'

Answers to Exercises

continued

Page 62

1.
```
select country
from religions
where religion='protestant'
and percent>40
union
select country
from countries
where language='german'
```

   There are 10 rows in the result.

2.
```
select country
from persons
where job='s'
union
select country
from armies
where budget>10000
order by 1
```

3.
```
select name
from persons
where name like '_z%'
union
select country
from countries
where country like '_z%'
union
select language
from countries
where language like '_z%'
```

   There are 9 rows in the result.

Page 65

1.
```
select max(pop)
from countries

select *
from countries
where pop=
(select max(pop)
from countries)
```

   China wins.

2.
```
select *
from armies
where budget>
(select avg(budget)
from armies)
```

3.
```
select language,name,title,bdate
from persons as p,jobs as j,countries as c
where p.job=j.job
and p.country=c.country
and p.job=
(select job
from persons
where name='luther')
order by language,name
```

Answers to Exercises

continued

Page 67

1. select country
   from countries
   where literacy<50

   select *
   from persons
   where job='w'
   and country in
   (select country
   from countries
   where literacy<50)

2. select *
   from persons
   where country in
   (select country
   from religions
   where religion='catholic'
   and percent>95)

3. select *
   from persons
   where job='m'
   and country in
   (select country
   from countries
   where language='english')
   and country in
   (select country
   from armies
   where troops<
   (select avg(troops)
   from armies))

Page 69

1. select job,name
   from persons as p
   where bdate=
   (select max(bdate)
   from persons
   where job=p.job)

2. select *
   from persons as p
   where bdate=
   (select max(bdate)
   from persons
   where gender=p.gender)

   Both are from Liechtenstein, are royalty, were born in the first half of the year, and have virtually unpronounceable names.

3. select country,religion
   from religions as r
   where percent=
   (select max(percent)
   from religions
   where country=r.country)
   and country in
   (select country
   from countries
   where language='german')

Answers to Exercises

continued

Page 72

1. select *
   from jobs

   insert
   into jobs(job,title)
   values('A','Author')

2. select *
   from persons
   where person>=500

   insert
   into persons
   (person,name,bdate,gender,country,job)
   values(500,'name','bdate','gen','country','job')

3. insert
   into persons
   (person,name,bdate,gender,country,job)
   values(501,'name','bdate','gen','country','job')

Page 74

1. update persons
   set job='A'
   where job='w'

   select *
   from persons
   order by job

2. update persons
   set bdate='bdate',job='E'
   where person=500

   select *
   from persons
   where person=500

3. update persons
   set job='W'
   where job='a'

   select *
   from persons
   order by job

Answers to Exercises

continued

Page 76

1. delete
   from persons
   where person=501

   delete
   from persons
   where person=500

   select *
   from persons
   where person>=500

   ow!

2. select count(*)
   from persons
   where bdate like '17%'

   delete
   from persons
   where bdate like '17%'

   select count(*)
   from persons
   where bdate like '17%'

3. select count(*)
   from religions

   There should be 389 rows.

   delete
   from religions
   where name like '%orthodox%'

   select count(*)
   from religions

Page 78

1. rollback

   select * from jobs

   'A' should no longer be in the JOBS table.

   select count(*) from religions

2. delete
   from persons

   delete
   from armies

   select * from persons
   select * from armies

   They're gone!

   rollback

   select * from persons
   select * from armies

   They're back!

3. insert
   into persons
   (person,name,bdate,gender,country,job)
   values(500,'name','bdate','gen','country','job')

   commit

   rollback

   select * from persons where person=500

   You're still there!

   delete
   from persons
   where person=500

   commit

Answers to Exercises

continued

Page 81

1. create table scientists
   (
   name char(20),
   bdate date,
   gender char(6),
   country char(20)
)

2. insert
   into scientists(name,bdate,gender,country)
   values('name','bdate','gen','country')

   select *
   from scientists

3. drop table scientists

   select *
   from scientists

Page 83

1. create table scientists
   (
   name char(20),
   bdate date,
   gender char(6),
   country char(20)
)

2. insert
   into scientists(name,bdate,gender,country)
   (select name,bdate,gender,country
   from persons
   where job='s'
   and gender='female')

   There is 1 female scientist.

3. insert
   into scientists(name,bdate,gender,country)
   (select name,bdate,gender,country
   from persons
   where job='s'
   and gender='male')

   select count(*)
   from scientists

   There are now 29 scientists.

4. insert
   into scientists(name,bdate,gender,country)
   (select name,bdate,gender,country
   from persons
   where job='t')

   select count(*)
   from scientists

   There are 42 scientists in all.

5. drop table scientists

Answers to Exercises

continued

Page 87

1. create table theologians
   (
   name char(20),
   bdate date,
   gender char(6) not null default 'male',
   country char(20),
   check(gender in ('male','female')),
   primary key(name),
   foreign key(country),
   references countries(country)
)

   insert
   into theologians(name,bdate,gender,country)
   (select name,bdate,gender,country
   from persons
   where job='t')

2. insert
   into theologians(name,bdate,gender,country)
   values('name','bdate','gen','Prussia')

   insert
   into theologians(name,bdate,gender,country)
   values('name','bdate',null,'country')

3. drop table theologians

Page 89

1. create table regions
   (
   name char(10)
)

   insert into regions(name) values('north')
   insert into regions(name) values('south')
   insert into regions(name) values('east')
   insert into regions(name) values('west')

   insert into regions(name)
   (select name from regions)

   Execute the above about 14 times.

2. select name,count(*)
   from regions
   group by name

   This query should take noticeably long to execute (several seconds is enough). If not, double the number of rows (by inserting the table into itself) and try again.

3. create index rx
   on regions(name)

   The query should now run significantly faster because the index is used to collect the rows for each group.

4. drop index rx
   drop table regions

Answers to Exercises

continued

Page 91

1. create view iv
   (person,name,bdate,gender,country,job)
   as
   select *
   from persons
   where country='israel'

   select *
   from iv

2. insert
   into iv
   (person,name,bdate,gender,country,job)
   values(502,'name','bdate','gen','Israel','job')

   insert
   into iv
   (person,name,bdate,gender,country,job)
   values(503,'name','bdate','gen','USA','job')

   Your long-lost friend is inserted (into the underlying PERSON table), but is not visible in the view because he/she does not satisfy the conditions stated in the view definition.

3. create view ijv(name,bdate,country,title)
   as
   select name,bdate,country,title
   from iv,jobs
   where iv.job=jobs.job

4. drop view ijv
   drop view iv

   So long. Farewell. Aufwiedersein. Shalom.

Extra Credit

This column contains answers to important questions beyond the scope of this text.

1. Yes, people are evil. This is easily seen from a) Scripture, b) reason, and c) experience.

   a) Scripture. A quote from Jeremiah should suffice here: 'The heart is deceitful above all things, and desperately wicked.'

   b) Reason. If people are basically good — even if a significant majority are — then why do we lock up our belongings? Why do we need a police force? an army?

   c) Experience. Have you ever noticed how little kids 'figure out' how to lie without any training? Did you have to be taught to be selfish or lazy, or did it just 'come to you'?

2. No, basically bad people are not fit for Heaven; they would spoil the place.

3. The old self must die, and a new self be installed in its place. This is totally the work of God and is 'discovered' by the individual, much like a newborn baby discovers its hands, feet, etc.

4. The theological term is 'regeneration'; this is commonly called 'being born again'.

5. No, very few experience this new birth. In Noah's day, only 8 survived the flood. Lot and his two daughters were the only ones to escape the judgment of God on Sodom.

6. No, the 'end of the world' is quite some time off, but the current age of grace is rapidly coming to a close. Delay is inadvisable.

7. The entire plan is described in the Bible. Our text, The Truth, is also yours for the taking.

Syntax Summary

SQL STATEMENTS BY CATEGORY

CATEGORY	STATEMENT	PURPOSE
Query	SELECT	Display rows of one or more tables
Maintenance	INSERT	Add rows to a table
	UPDATE	Change rows in a table
	DELETE	Remove rows from a table
Definition	CREATE	Add tables, indices, views
	DROP	Remove tables, indices, views

VALUES BY CATEGORY

CATEGORY	DESCRIPTION	EXAMPLES
NUMERIC	positive values	3, +12
	negative values	-7, -1024000
	decimal values	3.141519, -.96
NON-NUMERIC	single words	'Chamberlin', 'SELECT'
	multiple words	'We love SQL', 'The LORD is good to me'
	single quotes	'10 O''Clock', 'I don''t know'
DATE	'yyyy-mm-dd' format	'1996-01-01', '1996-12-31'

Syntax Summary

continued

COMPARISON OPERATORS

OPERATOR	MEANING	EXAMPLE
=	Equal to	NAME = 'EINSTEIN'
<>	Not equal to	BDATE <> '1944-05-02'
<	Less than	POP < 100000
<=	Less than or equal to	NAME <= 'O"Grady'
>	Greater than	AREA > 999
>=	Greater than or equal to	BDATE >= '1962-06-19'

ARITHMETIC OPERATORS

OPERATOR	MEANING	EXAMPLE
+	Add	2 + 2
-	Subtract	BDATE - 365
*	Multiply	POP * 1.25
/	Divide	PERCENT / 100
( )	Precedence	2 + (4 / 2)

STATISTICAL FUNCTIONS

FUNCTION	MEANING	EXAMPLE
COUNT ( )	Count all rows	COUNT (*)
	Count non-null rows	COUNT (JOB)
	Count unique rows	COUNT (DISTINCT JOB)
SUM ( )	Total value	SUM (POP)
MIN ( )	Smallest value	MIN (POP)
MAX ( )	Largest value	MAX (POP)
AVG ( )	Average value	AVG (POP / AREA)

Syntax Summary

continued

SELECT STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
SELECT	*	*
[DISTINCT]	col [AS alias] list	NAME, PERSONS.JOB, TITLE AS POS
	expr [AS alias] list	POP / AREA AS DENSITY
	func [AS alias] list	MIN (POP) AS LOWEST
FROM	table [AS alias] list	PERSONS, JOBS AS J
WHERE	col oper value	AREA > 3000000
[NOT]	col LIKE pattern	NAME LIKE '%ST_ _N%'
	cmpr AND cmpr	NAME LIKE '%ST_ _N%' AND JOB = 'S'
	col BETWEEN i AND j	LITERACY BETWEEN 55 AND 60
	cmpr OR cmpr	NAME = 'LUTHER' OR NAME = 'CALVIN'
	col IN (value list)	NAME IN ('POE', 'HUGO', 'DAHL')
	col IS NULL	POP IS NULL
	(compound cmpr)	(JOB = 'S' OR JOB = 'W')
	expr oper value	POP / AREA > 300
	col oper (subquery)	POP > (SELECT AVG (POP) ...)
	col IN (subquery)	JOB IN (SELECT DISTINCT JOB ...)
GROUP BY	col list	JOB, COUNTRY
HAVING	comparisons with funcs	COUNT (*) > 30
UNION	query	SELECT ...
ORDER BY	col [DESC] list	LANGUAGE DESC, COUNTRY
	pos [DESC] list	1 DESC, 3
	expr [DESC] list	POP / AREA DESC
	func [DESC] list	COUNT (*) DESC

Syntax Summary

continued

INSERT STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
INSERT		
INTO	table (col list)	COUNTRIES (COUNTRY, POP)
VALUES	(value list)	('Beulah', 144000)
	(query)	(SELECT ...)

UPDATE STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
UPDATE	table	COUNTRIES
SET	new value list	AREA = 7992, LANGUAGE = 'Hebrew'
WHERE	comparisons	COUNTRY = 'Beulah'

DELETE STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
DELETE		
FROM	table	COUNTRIES
WHERE	comparisons	COUNTRY = 'Yugoslavia'

ROLLBACK STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
ROLLBACK		

COMMIT STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
COMMIT		

Syntax Summary

continued

CREATE STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
CREATE	TABLE table	TABLE WRITERS
(		
	col type (size) list	NAME CHAR (20), BDATE DATE, SALES NUMERIC (9, 2)
	NOT NULL	NOT NULL
	DEFAULT value,	DEFAULT 0,
UNIQUE	(col list),	(NAME),
CHECK	(comparison),	(GENDER IN ('Male', 'Female')),
PRIMARY KEY	(col list),	(NAME),
FOREIGN KEY	(col list) REFERENCES table (col list)	(COUNTRY) REFERENCES COUNTRIES (COUNTRY)
)		

CREATE	INDEX index	INDEX GJX
ON	table (col list)	PERSONS (GENDER, JOB)

CREATE	VIEW view (col list)	VIEW PTV (PERSON, NAME, TITLE)
AS	query	SELECT ...

DROP STATEMENT

CLAUSE	PARAMETERS	EXAMPLE
DROP	TABLE table	TABLE WRITERS
	INDEX index	INDEX GJX
	VIEW view	VIEW PTV

A Brief Critique

In spite of its popularity and acceptance as an official standard, the SQL language is rather poorly defined and suffers from a number of internal inconsistencies, such as:

- The SELECT clause precedes the FROM clause, when, logically, one chooses the appropriate tables first, then decides which columns should be displayed.

- The DISTINCT keyword operates on rows, yet must be entered in the SELECT clause (where columns, not rows, are listed).

- The asterisk is used for multiple purposes, unnecessarily complicating the SQL syntax. Consider, for example, SELECT *, POP*1.1, and SELECT COUNT(*).

- Functions cannot be arbitrarily nested. For example, AVG(POP/AREA) is allowed, but AVG(COUNT(RELIGION)) is not.

- SQL ignores null values when applying statistical functions like AVG, rather than providing an option to either ignore them or include them in the row count.

- The GROUP BY clause should be optional since the required information can usually be deduced from the SELECT clause.

- The HAVING clause is confusing; statistical functions should be allowed in the WHERE clause, and the HAVING clause should be eliminated altogether.

- The syntax for different types of SQL statements is inconsistent. For example:

SELECT columns	INSERT	UPDATE table	DELETE
FROM tables	INTO table(columns)	SET column=value,...	FROM table
WHERE...	VALUES(values)	WHERE...	WHERE...

- View definitions may not contain ORDER BY clauses.

While some of these limitations are addressed in certain SQL dialects, that is exactly the problem — there shouldn't be dialects, there should be ONE good standard.

Recommended Reading

An Introduction to Database Systems

C. J. Date

Probably the best all-around book on relational database ever written. Useful as an introductory survey, but equally good as a reference text. We highly recommend this book to all who are interested in relational database technology.

A Guide to the SQL Standard

C. J. Date

Another excellent work from Mr. Date. Concise, thorough, yet relatively easy to read. Date has also authored or co-authored several other books on specific versions of SQL ('A Guide to DB2,' for example). These may be more meaningful to you if you know which particular product you will be using.

ERA and LOGIC Workshop Manuals

Relational Systems Corporation

Two great database design methodologies (even if we do say so ourselves) for quickly and accurately designing a relational database that can be immediately implemented on any SQL system. ERA is preferred for operational systems, while LOGIC is better for data warehouse applications. You should really invest in both.

The Holy Bible

God

User manual for all the Sons of Adam. Pure, tried, sure. To be read on a daily basis (bread alone won't do, as you should know). A lamp to our feet; a light to our path. More to be desired than gold, yea, than much fine gold. Also explains why human languages — even artificial ones like SQL — are so messy. See Genesis 11.

Index