SQL Programming

Comparison Operations

Page A-1: Intro

Up until now, the queries that we've written against the database only allowed us to be selective in choosing which columns (or fields) of information we wanted to see.

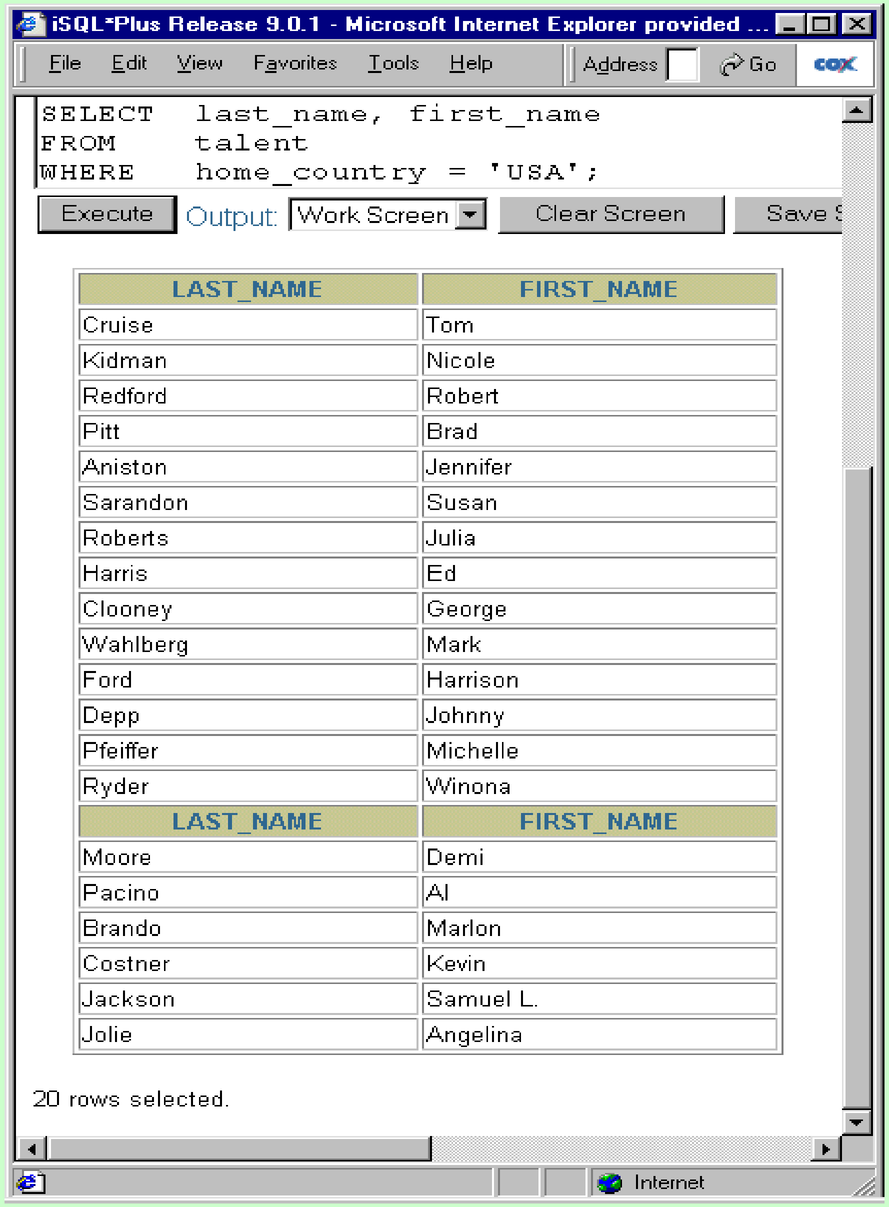
Now we'll look at some techniques that allow us some selectivity in the rows that are displayed.

This ability to select rows is the primary function of the WHERE clause.

Page B-1 WHERE Clause

The WHERE clause allows us to specify some condition that SQL will use in evaluating whether or not a row of information from the base table will be carried over to the result table.

Module 03: WHERE Clause Page B-2 WHERE Clause - Example


iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided ...
File Edit View Favorites Tools Help
Address
Go
Execute
Output: Work Screen
Clear Screen
Save
Internet

```
SELECT   last_name, first_name
FROM     talent
WHERE    home_country = 'USA';
```

LAST_NAME	FIRST_NAME
Cruise	Tom
Kidman	Nicole
Redford	Robert
Pitt	Brad
Aniston	Jennifer
Sarandon	Susan
Roberts	Julia
Harris	Ed
Clooney	George
Wahlberg	Mark
Ford	Harrison
Depp	Johnny
Pfeiffer	Michelle
Ryder	Winona
LAST_NAME	FIRST_NAME
Moore	Demi
Pacino	Al
Brando	Marlon
Costner	Kevin
Jackson	Samuel L.
Jolie	Angelina

```
20 rows selected.
```

Consider the following SQL program:

```
SELECT   last_name, first_name
FROM     talent
WHERE    home_country = 'USA';
```

In this example the WHERE clause applies the following test against each row in the base table as it evaluates them for inclusion in the result table:

Does the home_country column,

in this row of data, contain the value 'USA'?

Each row that passes this test is included in the result table. Rows that do not pass the test do not contribute any information to the result table.

Page B-3 Conditions

The WHERE clause is the place where the programmer lists all of the conditions that must be met before a row is included in the result table.

Here are some examples of the kinds of conditions that might be specified for our TALONS case study.

Is the last name Schwarzenegger?
Is the first name Angelina?
Does the first name start with a vowel?
Is the home state on the west coast?
Is this person (talent) younger than 21 yrs?

Page B-4: Predicate Conditions

These ‘English language’ questions need to be rephrased in a format that SQL can understand.

And it’s up to you, the SQL programmer, to do this translation.

The conditions must be written according to the rules of SQL predicate expressions.

One of these SQL rules insists that the evaluation of every predicate expression must result in one of these answers:

TRUE
FALSE
or UNKNOWN

Page B-5: Unknown???

In one of the previous modules I warned you that some SQL behavior is different from that of other programming languages, and this is one area where experienced computer programmers have a little difficulty.

Most programming languages use a 2-state logic that permits only 2 answers in the evaluation of predicate expressions: TRUE or FALSE. SQL uses a 3-state logic that permits 3 answers: TRUE, FALSE, or UNKNOWN.

Consider the question: Is this person older than 21 yrs? To answer this question we'd take today's date and subtract the person's birth date and evaluate the result. If the result is more than 21 the answer is TRUE, otherwise the answer is FALSE.

How can there be any answer other than TRUE or FALSE?

Page B-6: NULLs

What happens when you don't know the birth date? The only possible answer in that case is: I can't answer the question, I don't know how old they are. Hence the answer is UNKNOWN.

In almost every other programming language variables (fields) must be assigned some value. Sometimes programmers use codes to represent unknown data. They'll leave a name field BLANK, or they'll set the HOURLY WAGE to 0 (zero) if they don't know the value.

SQL is unique in that it allows the use of a special value known as NULL. We use NULL whenever we don't know what the value should be.

So, in the evaluation of predicate expressions, whenever SQL encounters a field of data that contains this NULL value, the answer will usually be UNKNOWN.

Page B-7: NULLs (cont)

We'll have a lot more to say about NULLs much later in the course.

For now, just be aware that

1. NULL values exist
2. SQL uses a special 3-state logic unlike most other programming languages
3. The odd thing about this 3-state logic is the possibility of an UNKNOWN answer
4. UNKNOWN answers are related to NULL values

If this topic is a little confusing, again, don't worry. We'll discuss it again before you need to deal with it in any programming problems.

Page B-8: Reality Check

How are you doing with the material?

I’m aware that I just introduced a whole bunch of terminology and I’m afraid I’m making this more complex than it needs to be.

So. Let’s take a break from the theoretical discussions and look at a few examples.

Page C-1: Problem 4-1 Statement

Our user community needs a report …

Here's what they want:

Of the clients we represent, which of them has experience working in theater?

That's a little vague.

Here's what I know about the database. The column named THEATRE includes a YES value for our clients who have acted in theater.

I can use this column to select the records that should be displayed, but I don't know what information the users want to see in the report. Do they want to see the client's id, their last name, both of their names, …

I tell the analyst I need more precise specifications and he returns with this:

User Need Rephrased:

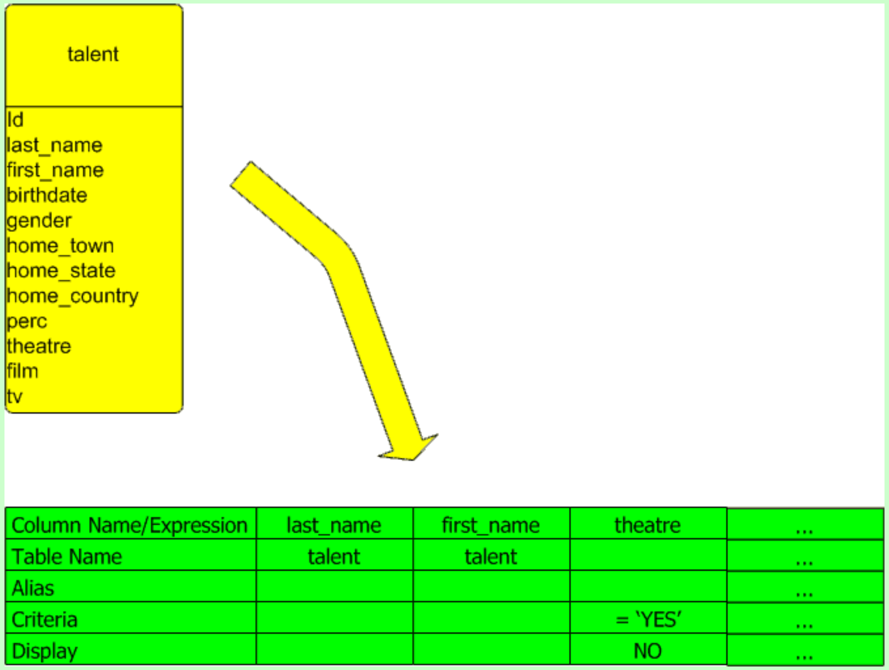
Prepare a report showing last name and first name of all of our clients where the THEATRE field in their record has a YES value.

Page C-2: Problem 4-1 Design and Code

Step 1: Build the Table Build Chart (TBC)

Step 2: Double check your TBC solution

Step 3: Transform the TBC into code.

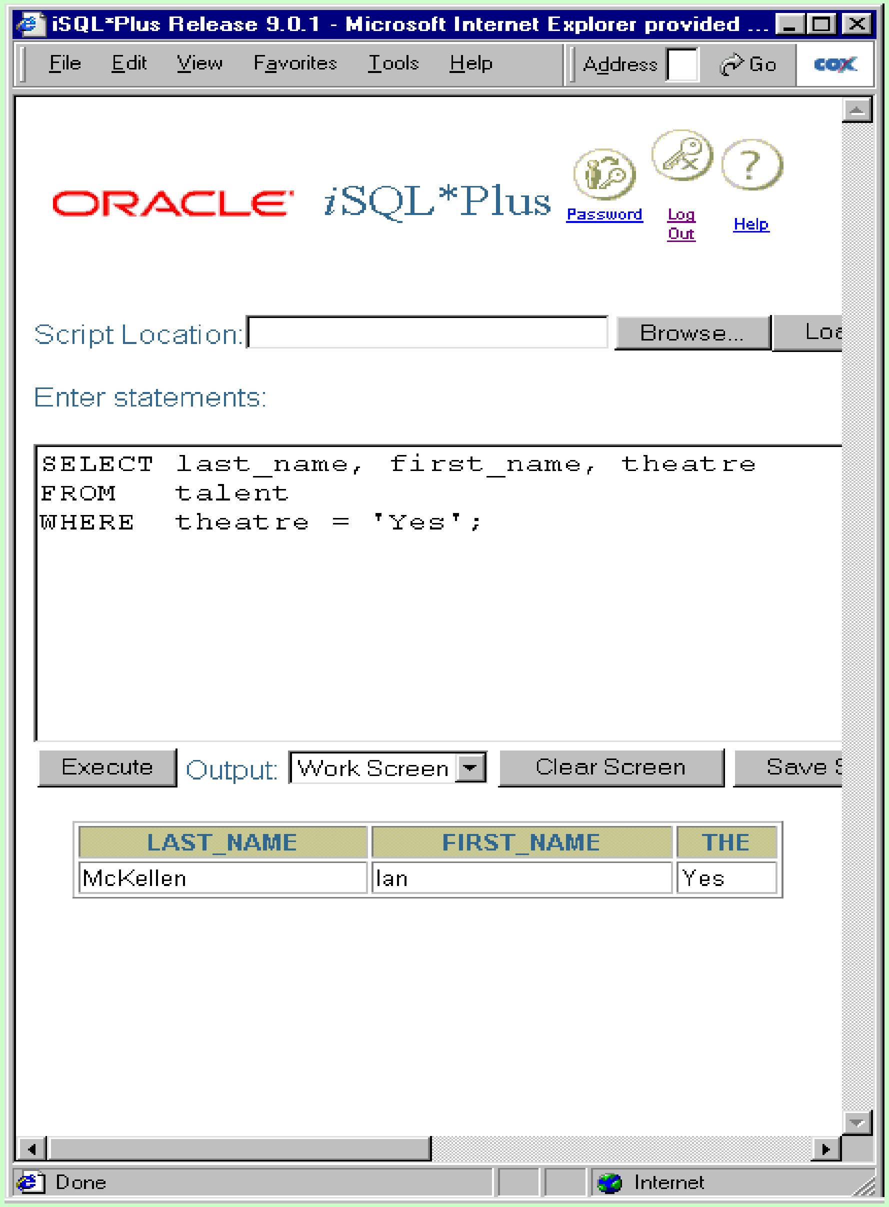


Column Name/Expression	last_name	first_name	theatre	...
Table Name	talent	talent		...
Alias				...
Criteria			= 'YES'	...
Display			NO	...

Notice that the specification of the predicate condition is placed in the criteria row of the TBC. Also note that it has been *rephrased* - it's not written in the format of an English statement, it looks *computer-ish*.

```
SELECT  last_name, first_name, theatre
FROM     talent
WHERE    theatre = 'Yes';
```

Module 03: WHERE Clause   Page C-3: Problem 4-1 Solution

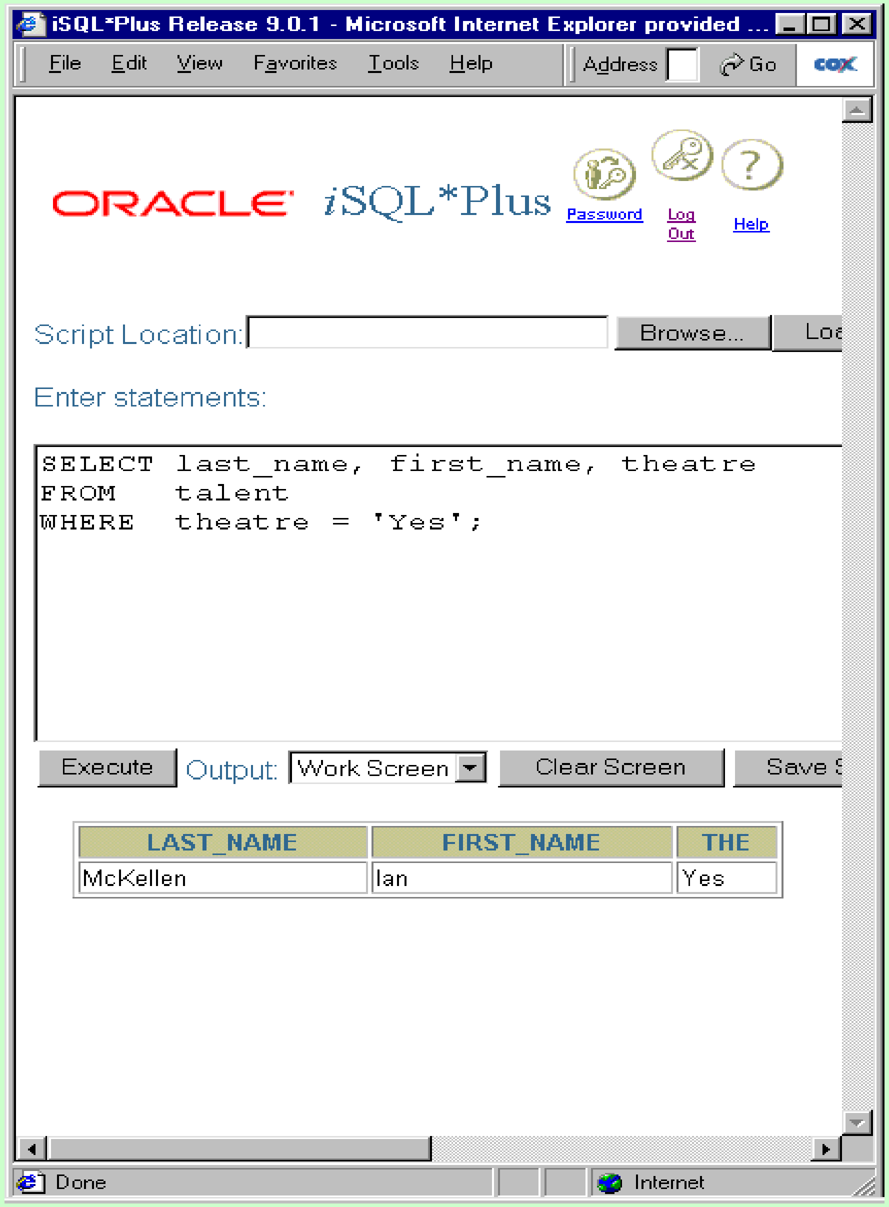


```
SELECT last_name, first_name, theatre
FROM   talent
WHERE  theatre = 'Yes';
```

LAST_NAME	FIRST_NAME	THE
McKellen	Ian	Yes

Here's the result, and upon analysis it appears that we have only a single client who has experience working in theatre.

Module 03: WHERE Clause Page C-4: Problem 4-1 Commentary



```
SELECT last_name, first_name, theatre
FROM   talent
WHERE  theatre = 'Yes';
```

LAST_NAME	FIRST_NAME	THE
McKellen	Ian	Yes

The ordering of clauses is:

SELECT
FROM
WHERE

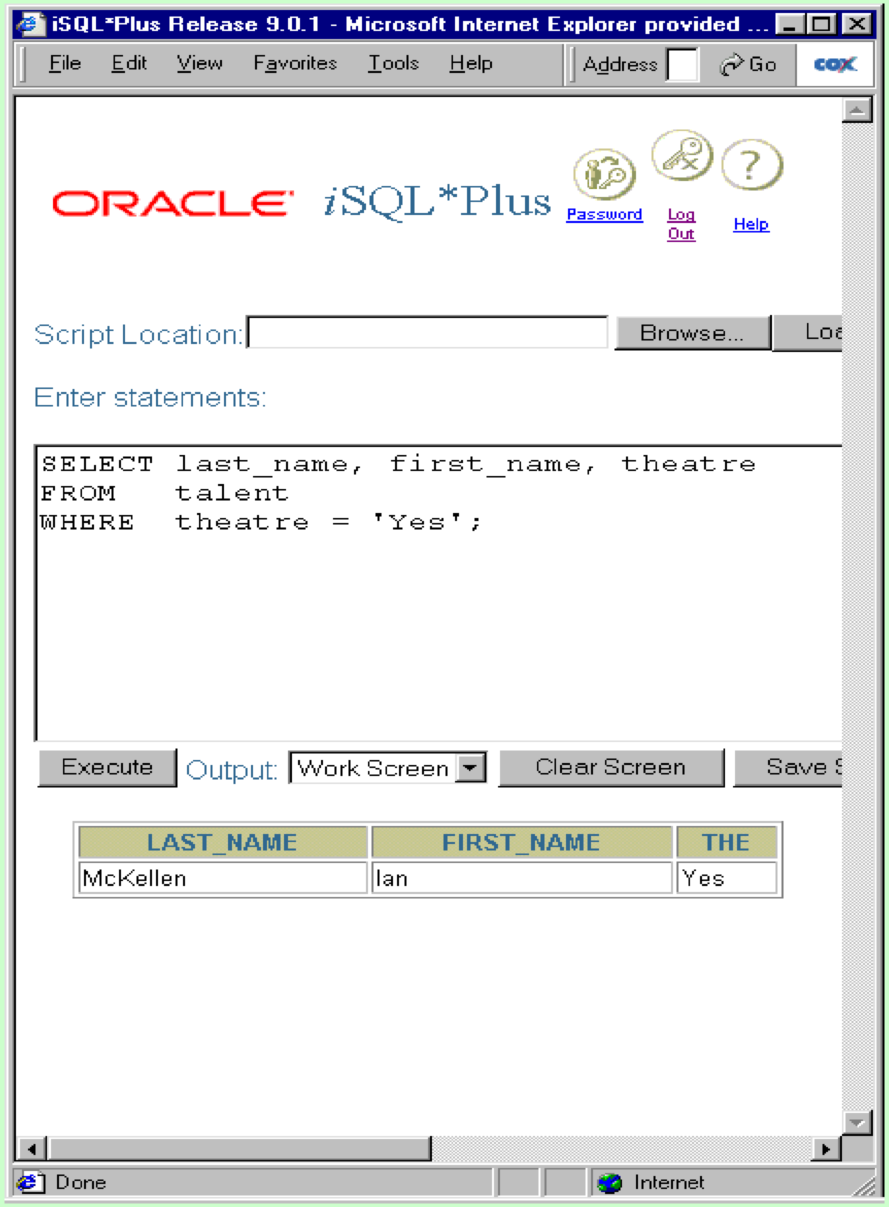
This particular predicate expression involves a simple comparison of 'equality'.

Use your English skills to decipher these tricky computing terms. An 'equality' comparison checks two items to see if they're EQUAL!!! (ie. Checks to see if they're the same).

The two items in this comparison are:

the theatre column in each row, and
the literal value 'Yes'.

Module 03: WHERE Clause   Page C-5: Programming Strategies



```
SELECT last_name, first_name, theatre
FROM   talent
WHERE  theatre = 'Yes';
```

LAST_NAME	FIRST_NAME	THE
McKellen	Ian	Yes

In the ‘real world’ when you prepare and submit a report to your user community, they expect the information to be correct.

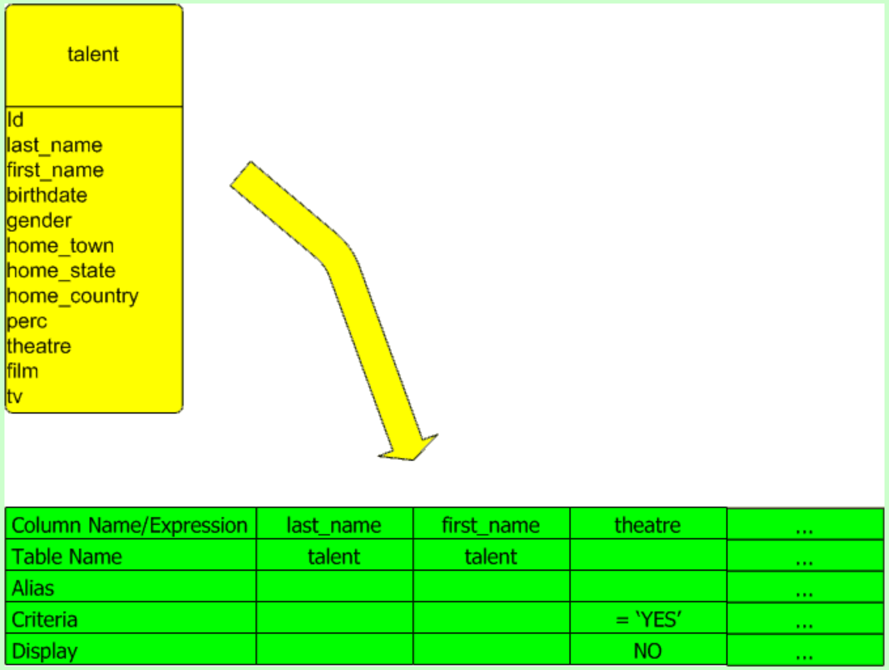
As the programmer you’ll need to develop intuitions about the database your company is using, so that when you run these reports you’ll have a gut feel as to whether or not the results are in the ballpark.

The user community didn’t ask for the THEATRE column to be displayed, but as the programmer, I want to see that data because it’s one of the criteria I’m using in my WHERE clause. I tend to display all of the important columns while I’m developing my SQL programs, and once the program is working correctly, then I pare things down to the user specifications.

Module 03: WHERE Clause Page C-6: Problem 4-2 Statement

Now that the program seems to be working
I’ll remove the extraneous fields.

Module 03: WHERE Clause Page C-7: Problem 4-2 Design and Code



Column Name/Expression	last_name	first_name	theatre	…
Table Name	talent	talent		…
Alias				…
Criteria			= 'YES'	…
Display			NO	…

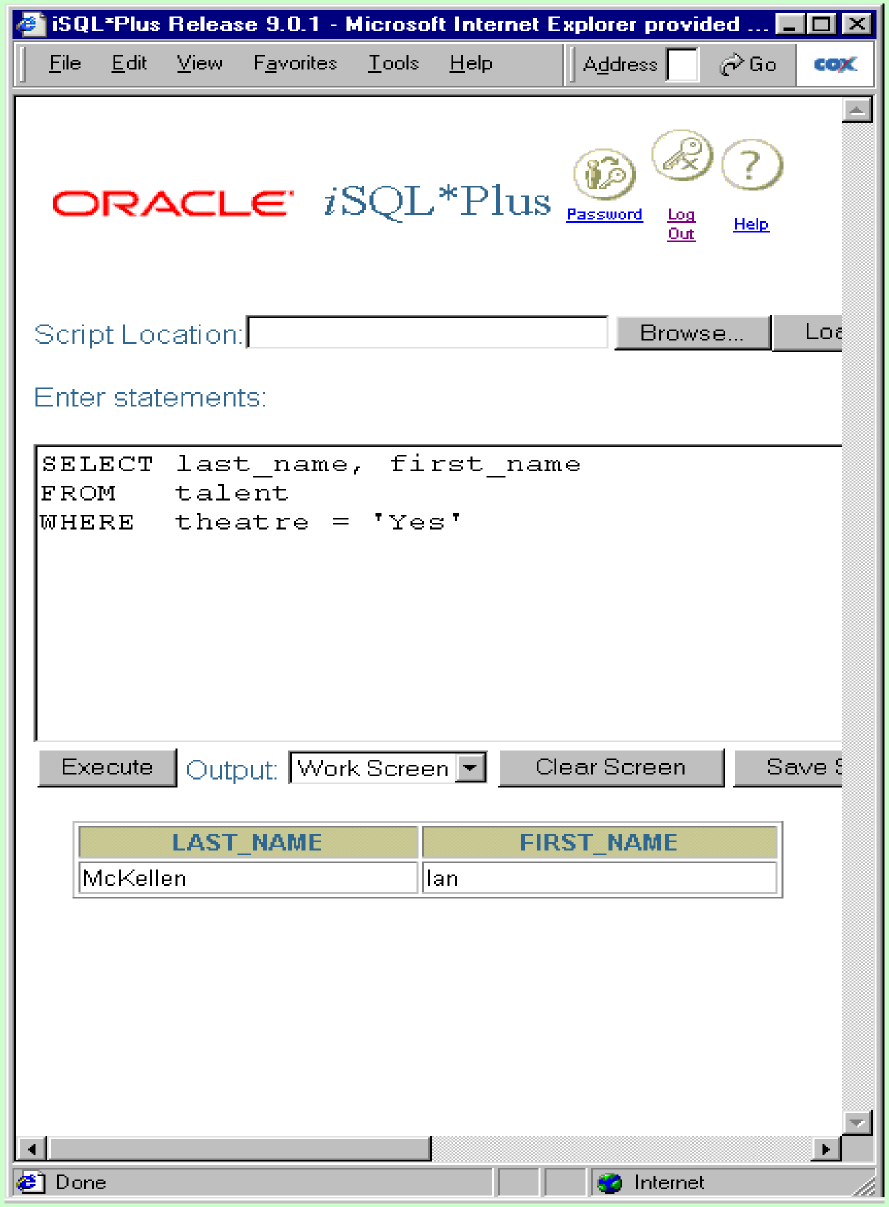
Here's what I've got,

```
SELECT  last_name, first_name, theatre
FROM    talent
WHERE   theatre = 'Yes';
```

And I'll remove the theatre column from the SELECT clause so that my code conforms to the user specifications

```
SELECT  last_name, first_name
FROM    talent
WHERE   theatre = 'Yes';
```

Page C-8: Problem 4-2 Solution

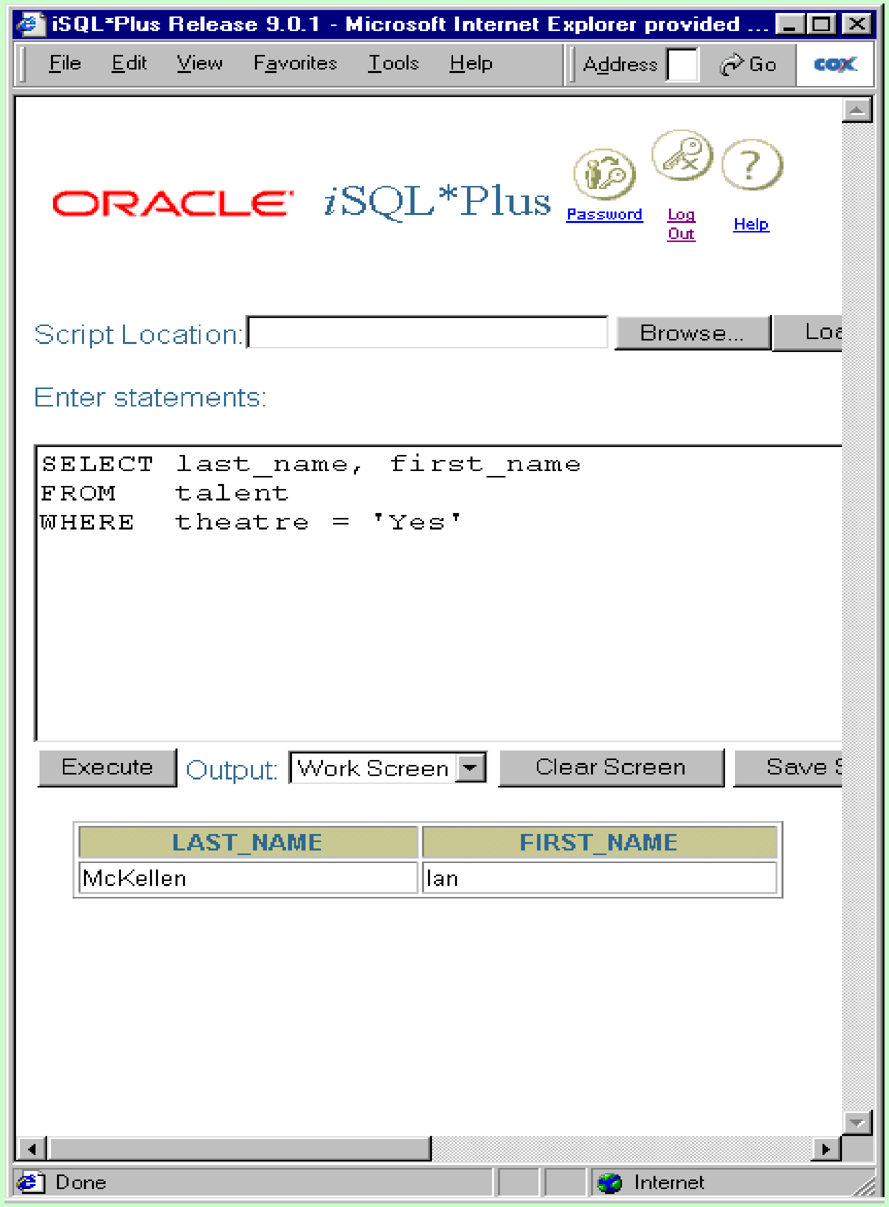


```
SELECT  last_name, first_name
FROM    talent
WHERE   theatre = 'Yes'
```

LAST_NAME	FIRST_NAME
McKellen	Ian

Here's the result, just as the users requested.

Page C-9: Problem 4-2 Commentary



```
SELECT last_name, first_name
FROM   talent
WHERE  theatre = 'Yes'
```

LAST_NAME	FIRST_NAME
McKellen	Ian

Here's something interesting to note.

The WHERE clause can reference any column in any of the base tables that are listed in the FROM clause, even if they are NOT listed in the SELECT clause.

Module 03: WHERE Clause Page C-10: Problem 4-3 Statement

The user wants to know which of our clients were not born in the US.

User needs rephrased:

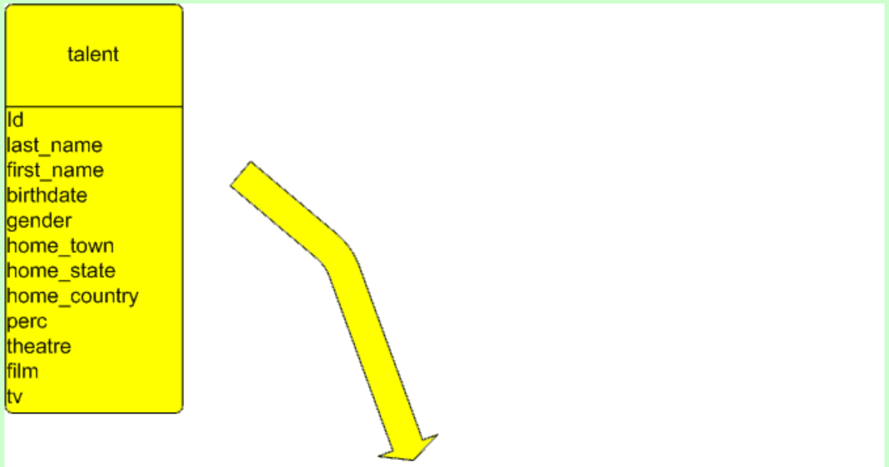
Prepare a report showing name information for all of our clients whose home country is not the US.

Module 03: WHERE Clause   Page C-11: Problem 4-3 Design & Code

Step 1: Build the Table Build Chart (TBC)

Step 2: Double check your TBC solution

Step 3: Transform the TBC into code.

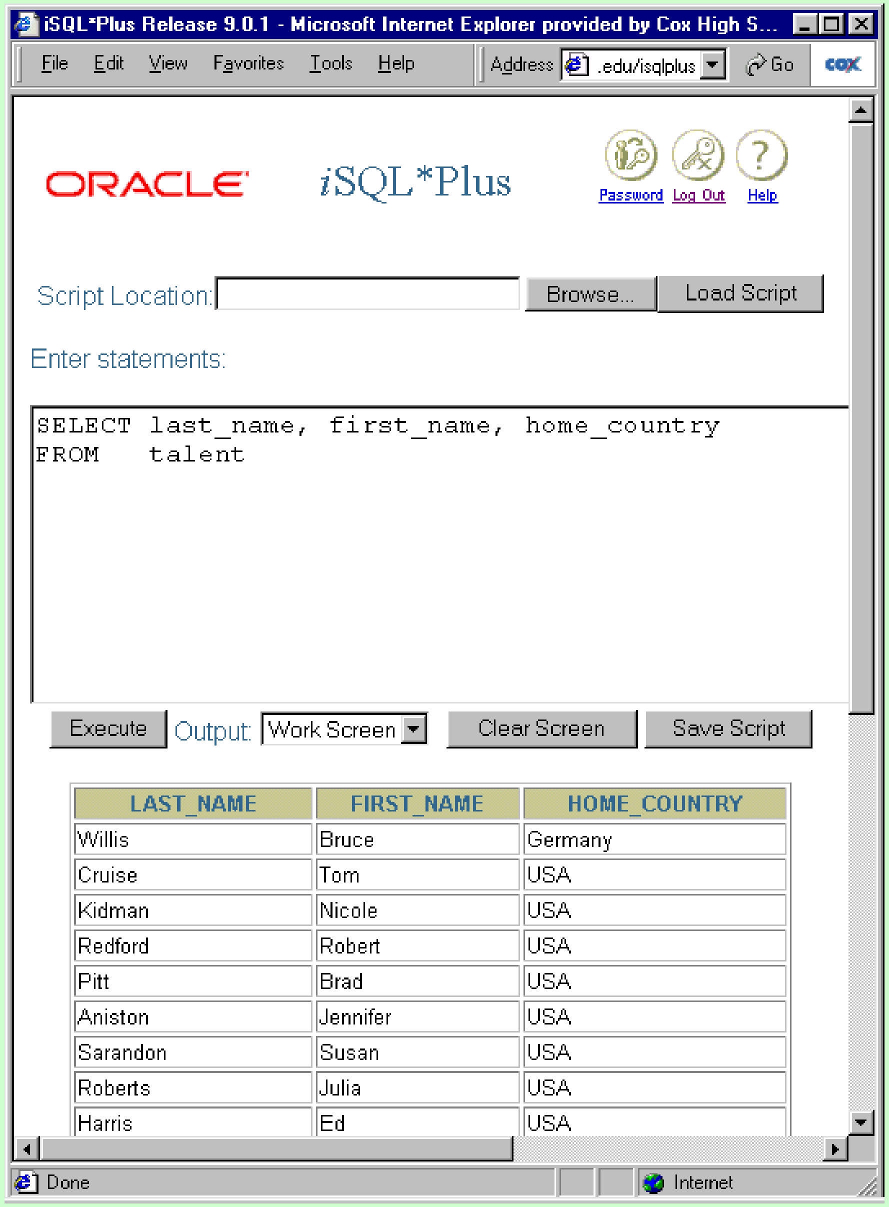


Column Name/Expression	last_name	first_name	home_country	...
Table Name	talent	talent		...
Alias				...
Criteria			<> 'US'	...
Display			NO	...

Ya know, they want to know about clients born in the US, but my recollection is that we use USA (not US) as the home_country value.

So, I'm going to double check that first.

Module 03: WHERE Clause Page C-12: Problem 4-3 Test

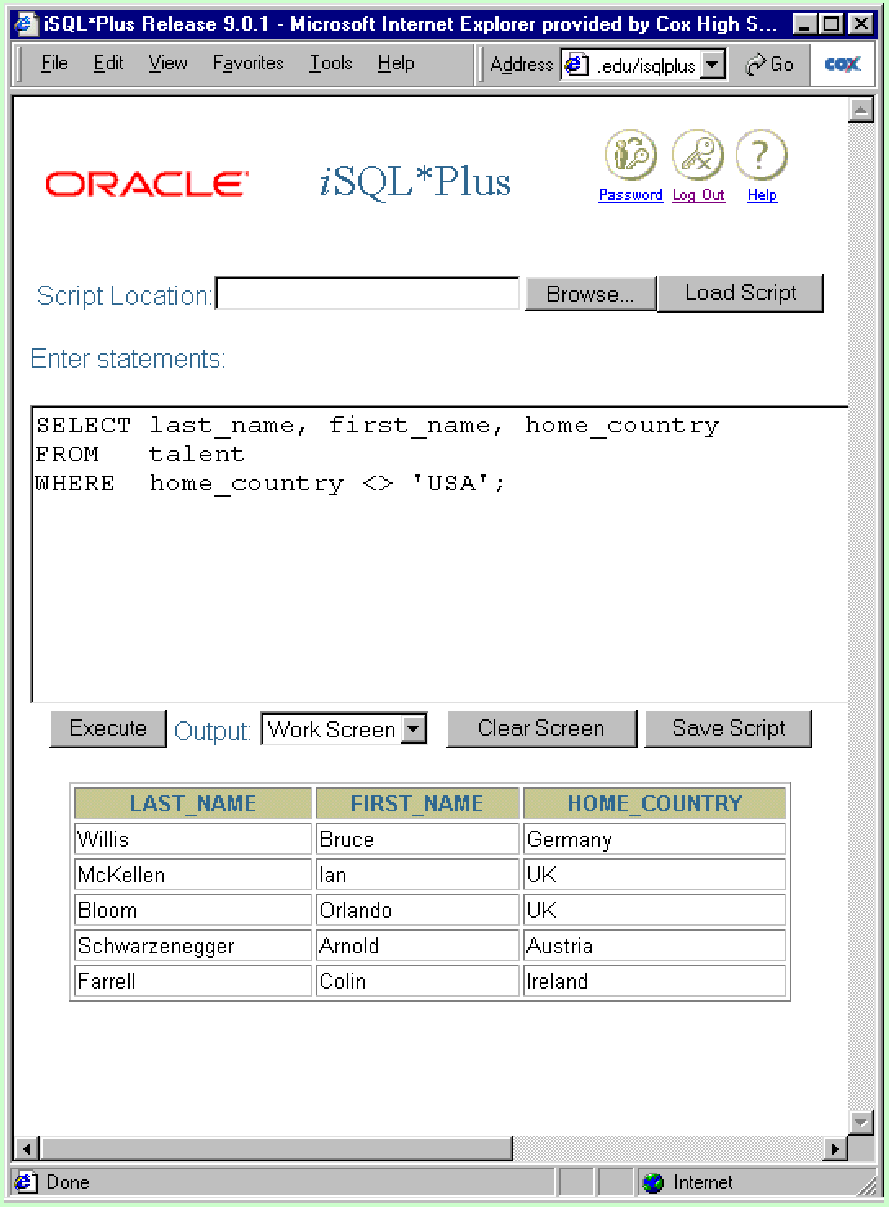

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High S...
File Edit View Favorites Tools Help
Address .edu/isqlplus
Go
ORACLE iSQL*Plus
Password Log Out Help
Script Location:
Browse... Load Script
Enter statements:
Execute Output: Work Screen Clear Screen Save Script
Done
Internet

```
SELECT last_name, first_name, home_country
FROM   talent
```

LAST_NAME	FIRST_NAME	HOME_COUNTRY
Willis	Bruce	Germany
Cruise	Tom	USA
Kidman	Nicole	USA
Redford	Robert	USA
Pitt	Brad	USA
Aniston	Jennifer	USA
Sarandon	Susan	USA
Roberts	Julia	USA
Harris	Ed	USA

Aha! Just as I suspected. I need to reformulate the code.

Module 03: WHERE Clause   Page C-13: Problem 4-3 Design & Code

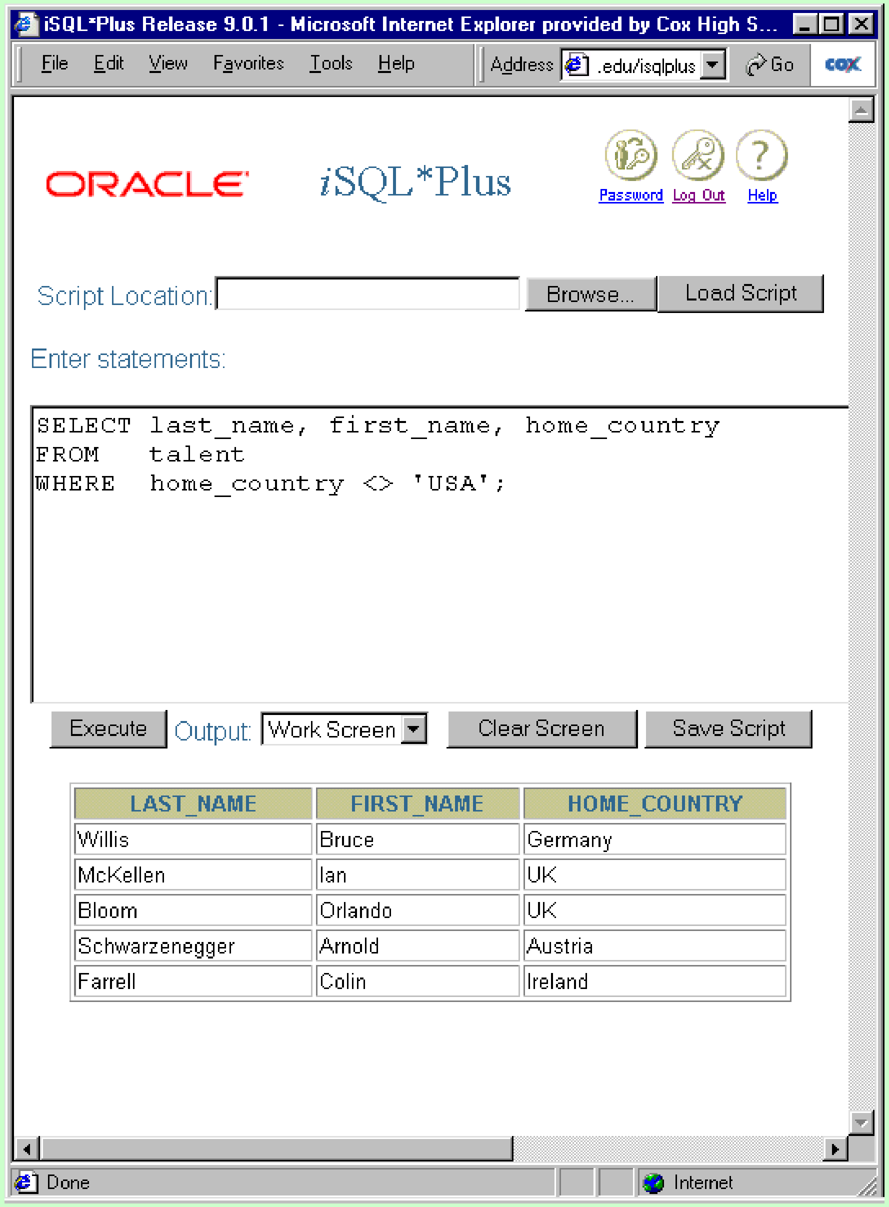

iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High S...
File Edit View Favorites Tools Help
Address .edu/isqlplus Go
ORACLE iSQL*Plus
Password Log Out Help
Script Location: Browse... Load Script
Enter statements:
Execute Output: Work Screen Clear Screen Save Script
Done Internet

```
SELECT last_name, first_name, home_country
FROM   talent
WHERE  home_country <> 'USA';
```

LAST_NAME	FIRST_NAME	HOME_COUNTRY
Willis	Bruce	Germany
McKellen	Ian	UK
Bloom	Orlando	UK
Schwarzenegger	Arnold	Austria
Farrell	Colin	Ireland

```
SELECT  last_name, first_name
FROM    talent
WHERE   home_country <>  'USA';
```

Module 03: WHERE Clause Page C-14: Problem 4-3 Commentary


iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High S...
File Edit View Favorites Tools Help
Address .edu/isqlplus Go
ORACLE iSQL*Plus
Password Log Out Help
Script Location: Browse... Load Script
Enter statements:
Execute Output: Work Screen Clear Screen Save Script
Done Internet

```
SELECT last_name, first_name, home_country
FROM   talent
WHERE  home_country <> 'USA';
```

LAST_NAME	FIRST_NAME	HOME_COUNTRY
Willis	Bruce	Germany
McKellen	Ian	UK
Bloom	Orlando	UK
Schwarzenegger	Arnold	Austria
Farrell	Colin	Ireland

The comparison operation that we've specified in this predicate expression is a test for inequality.

Notice that the inequality operator (the symbol) is <>.

Page C-15: Problem 4-4 Statement

Users need a report highlighting just those clients for whom we take more than an 8% cut.

User need rephrased

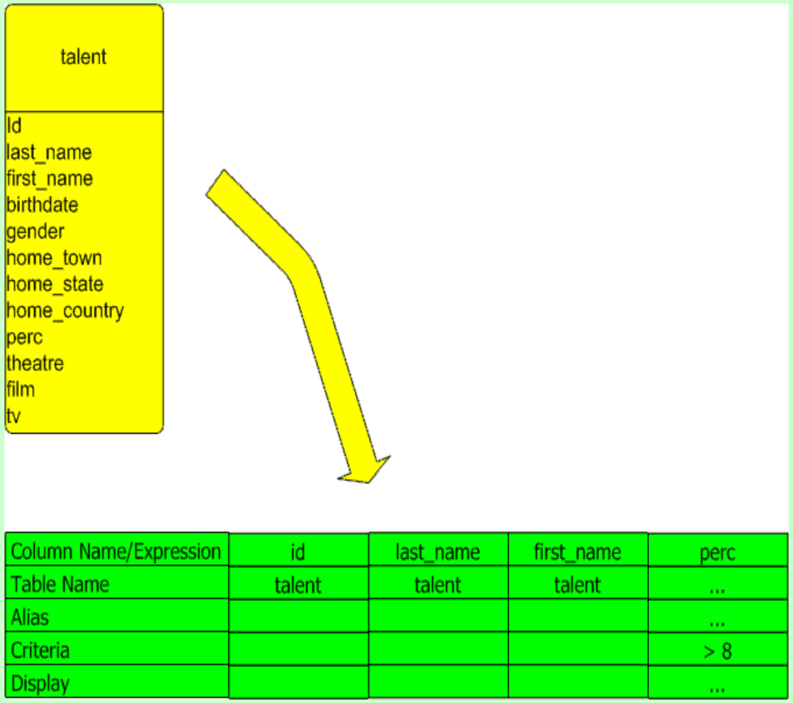
Prepare a report showing id and name information for all of our clients whose percentage field is greater than 8.

Module 03: WHERE Clause Page C-16: Problem 4-4 Design and Code

Step 1: Build the Table Build Chart (TBC)

Step 2: Double check your TBC solution

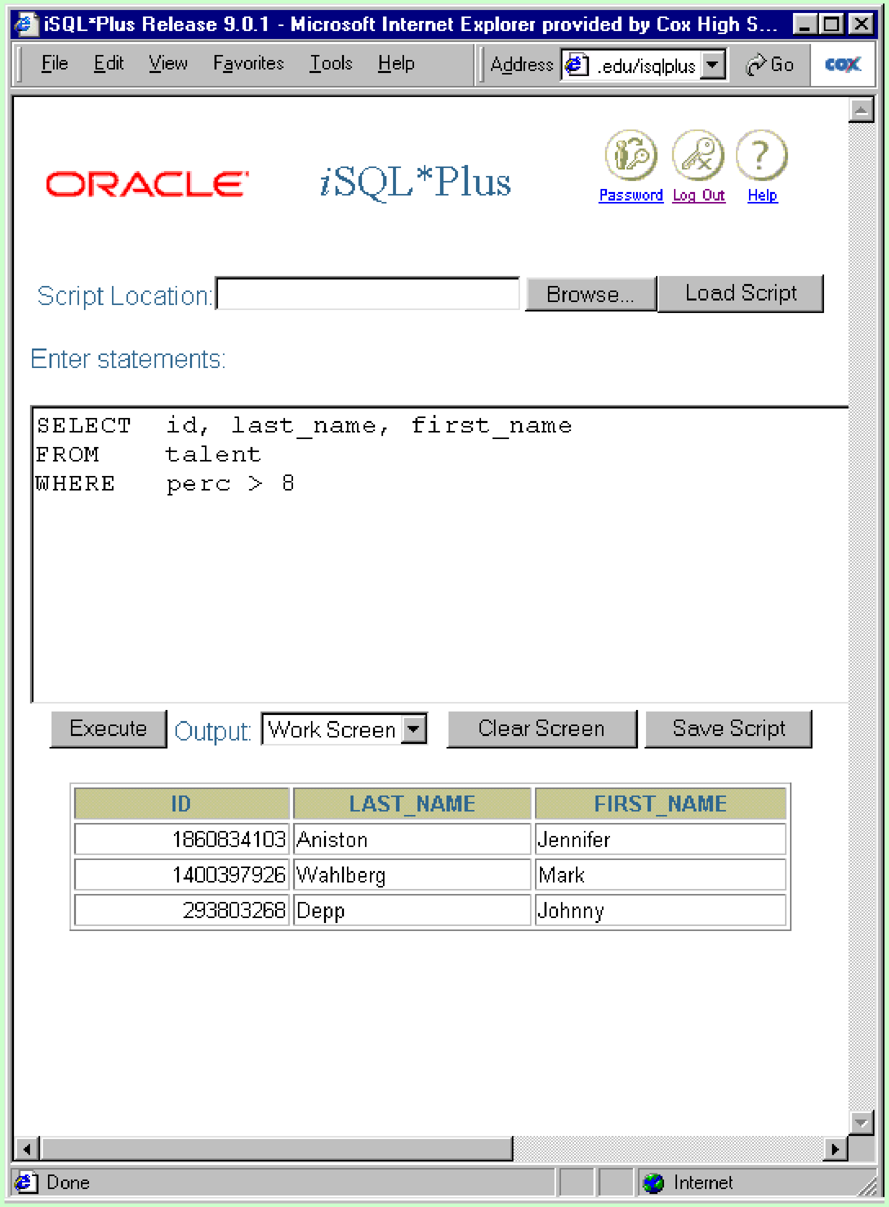
Step 3: Transform the TBC into code.



Column Name/Expression	id	last_name	first_name	perc
Table Name	talent	talent	talent	...
Alias				...
Criteria				> 8
Display				...

```
SELECT  id, last_name, first_name
FROM     talent
WHERE   perc > 8;
```

Module 03: WHERE Clause Page C-17: Problem 4-4 Solution

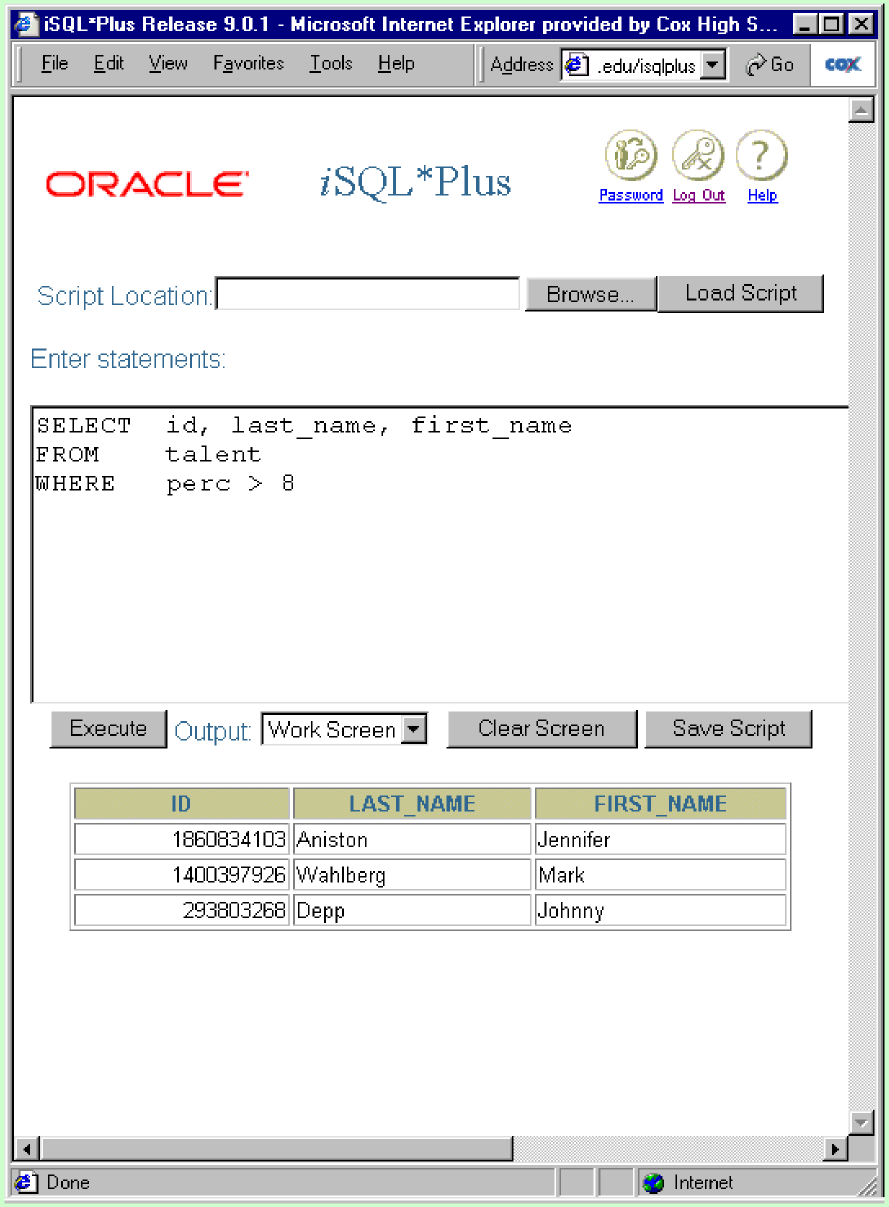


```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc > 8
```

ID	LAST_NAME	FIRST_NAME
1860834103	Aniston	Jennifer
1400397926	Wahlberg	Mark
293803268	Depp	Johnny

Here's the solution.

Module 03: WHERE Clause   Page C-18: Problem 4-4 Commentary


iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High S...
File Edit View Favorites Tools Help
Address .edu/isqlplus Go
ORACLE iSQL*Plus
Password Log Out Help
Script Location:
Browse... Load Script
Enter statements:
Execute Output: Work Screen Clear Screen Save Script
Done Internet

```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc > 8
```

ID	LAST_NAME	FIRST_NAME
1860834103	Aniston	Jennifer
1400397926	Wahlberg	Mark
293803268	Depp	Johnny

In this problem, the predicate expression applies a ‘greater than’ comparison.

The symbol, or operator, for the greater than condition is >

Page C-19: Problem 4-5 Statement

The last report highlighted the clients for whom we charge on the ‘high end’ of the scale.

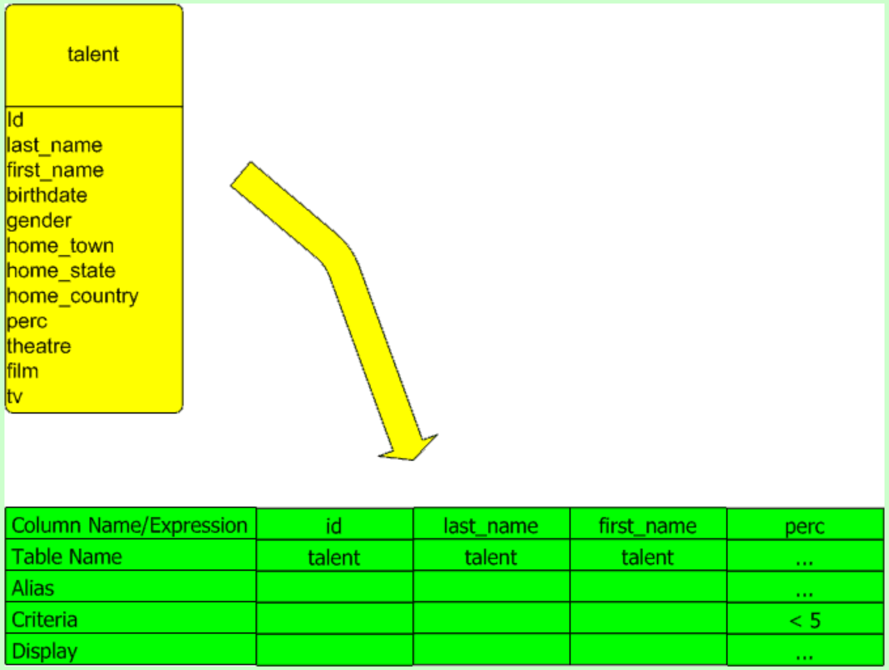
Now the users want to know about the clients on the other end of the scale.

User need rephrased

Prepare a report showing id and name information for all of our clients whose percentage field is less than 5.

Module 03: WHERE Clause Page C-21: Problem 4-5 Design & Code

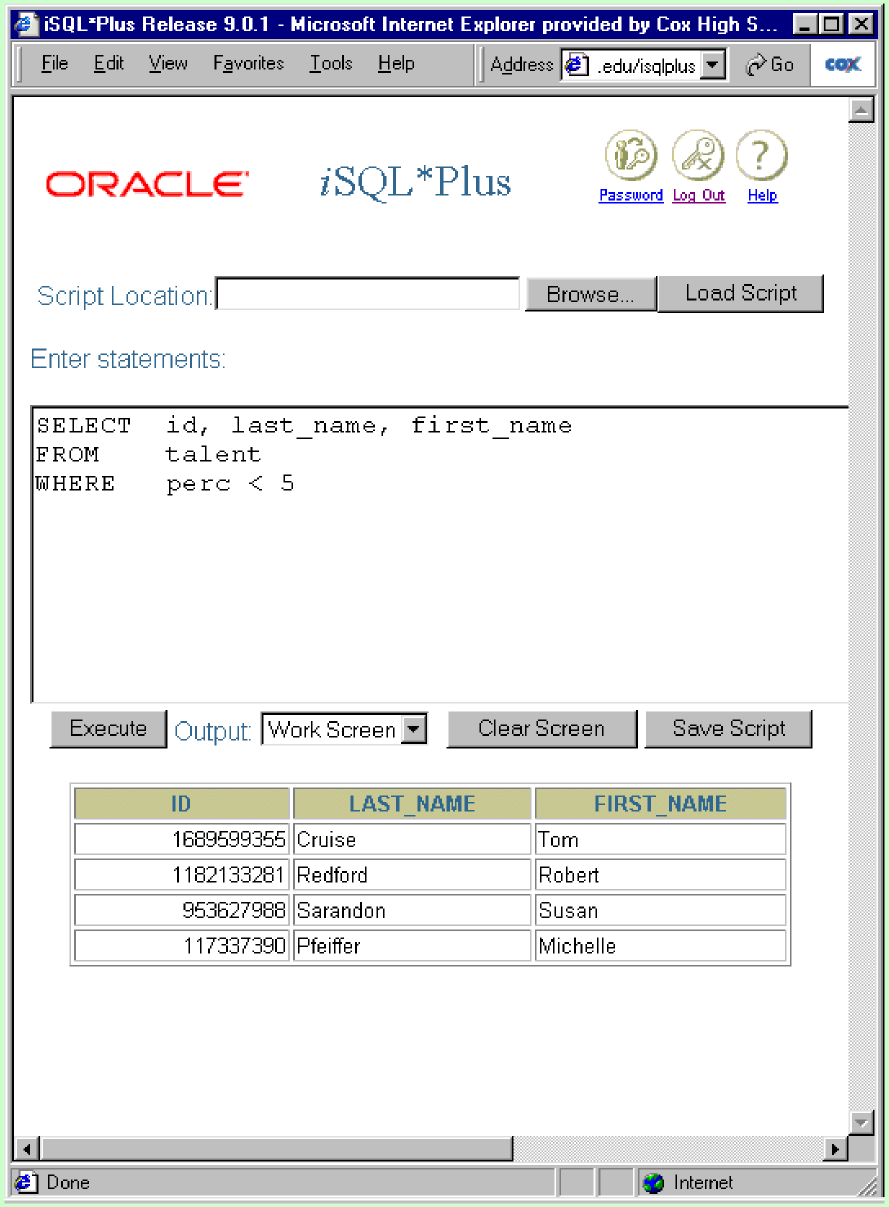
Step 1: Build the Table Build Chart (TBC)
Step 2: Double check your TBC solution
Step 3: Transform the TBC into code.



Column Name/Expression	id	last_name	first_name	perc
Table Name	talent	talent	talent	...
Alias				...
Criteria				< 5
Display				...

```
SELECT  id, last_name, first_name
FROM     talent
WHERE    perc < 5;
```

Module 03: WHERE Clause   Page C-21: Problem 4-5 Solution



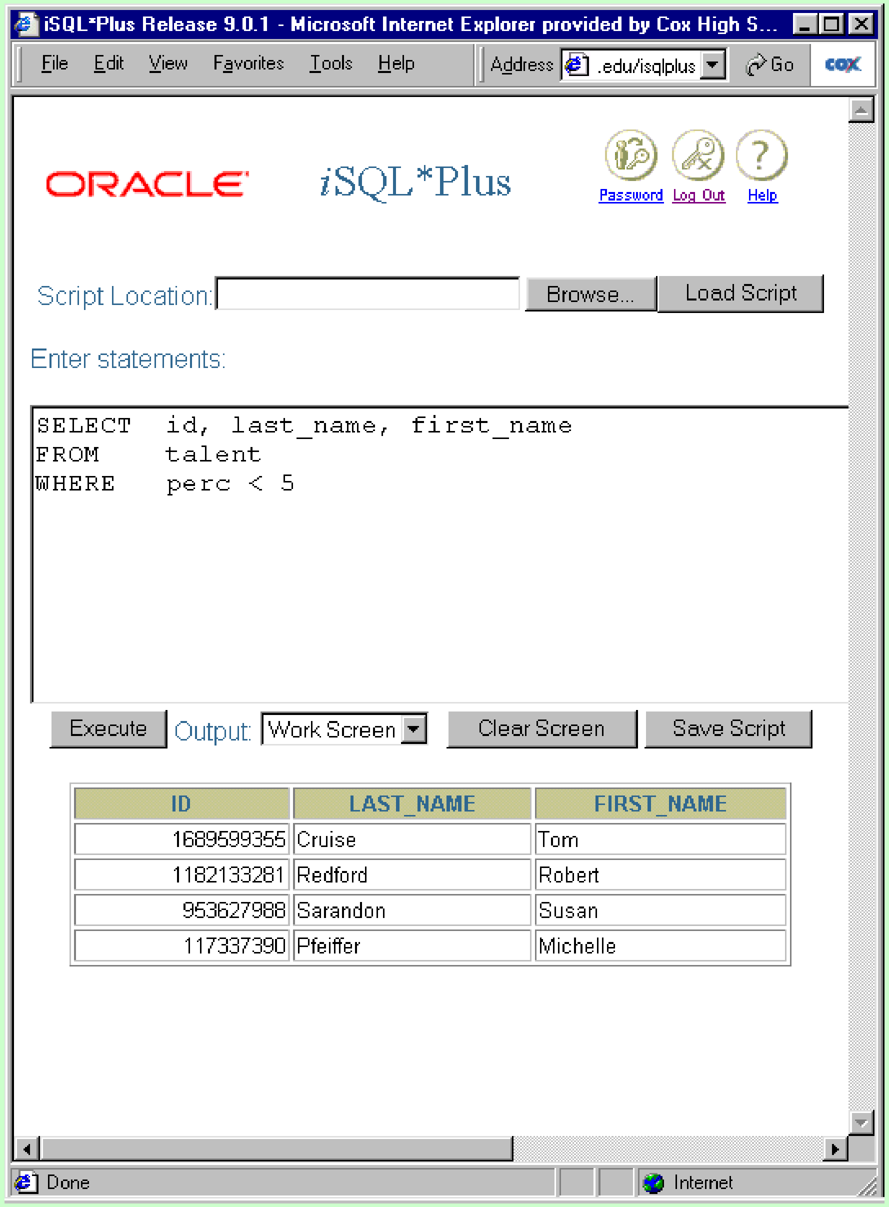
iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High S...
File Edit View Favorites Tools Help
Address .edu/isqlplus Go
ORACLE iSQL*Plus
Password Log Out Help
Script Location:
Browse... Load Script
Enter statements:
Execute Output: Work Screen Clear Screen Save Script
Done Internet

```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc < 5
```

ID	LAST_NAME	FIRST_NAME
1689599355	Cruise	Tom
1182133281	Redford	Robert
953627988	Sarandon	Susan
117337390	Pfeiffer	Michelle

Here’s the solution.

Module 03: WHERE Clause Page C-22: Problem 4-5 Commentary



```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc < 5
```

ID	LAST_NAME	FIRST_NAME
1689599355	Cruise	Tom
1182133281	Redford	Robert
953627988	Sarandon	Susan
117337390	Pfeiffer	Michelle

In this problem, the predicate expression applies a ‘less than’ comparison.

The symbol, or operator, for the less than condition is <.

Page C-23: Problem 4-6 Statement

In the next two sample problem we'll look at a slight variation on the 'greater than' and 'less than' comparisons.

The user community asks for a report showing which of our clients are charged 5% or less.

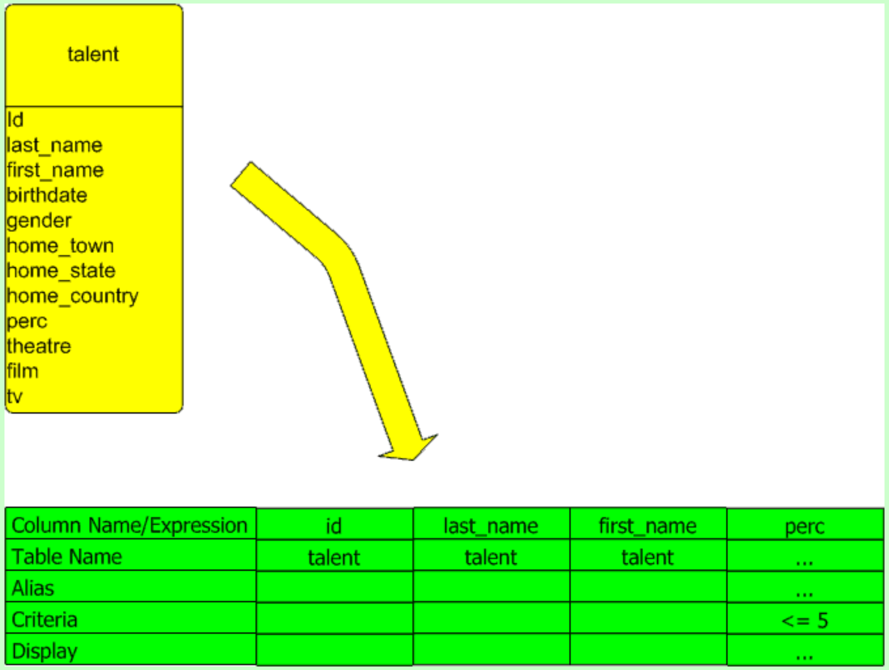
User need rephrased

Prepare a report showing id and name information for all of our clients whose percentage field is less than, or equal to 5.

Module 03: WHERE Clause   Page C-24: Problem 4-6 Design & Code

You won’t notice much of a change in either our methodology, or our solution. The only item that will change will be the comparison operator.

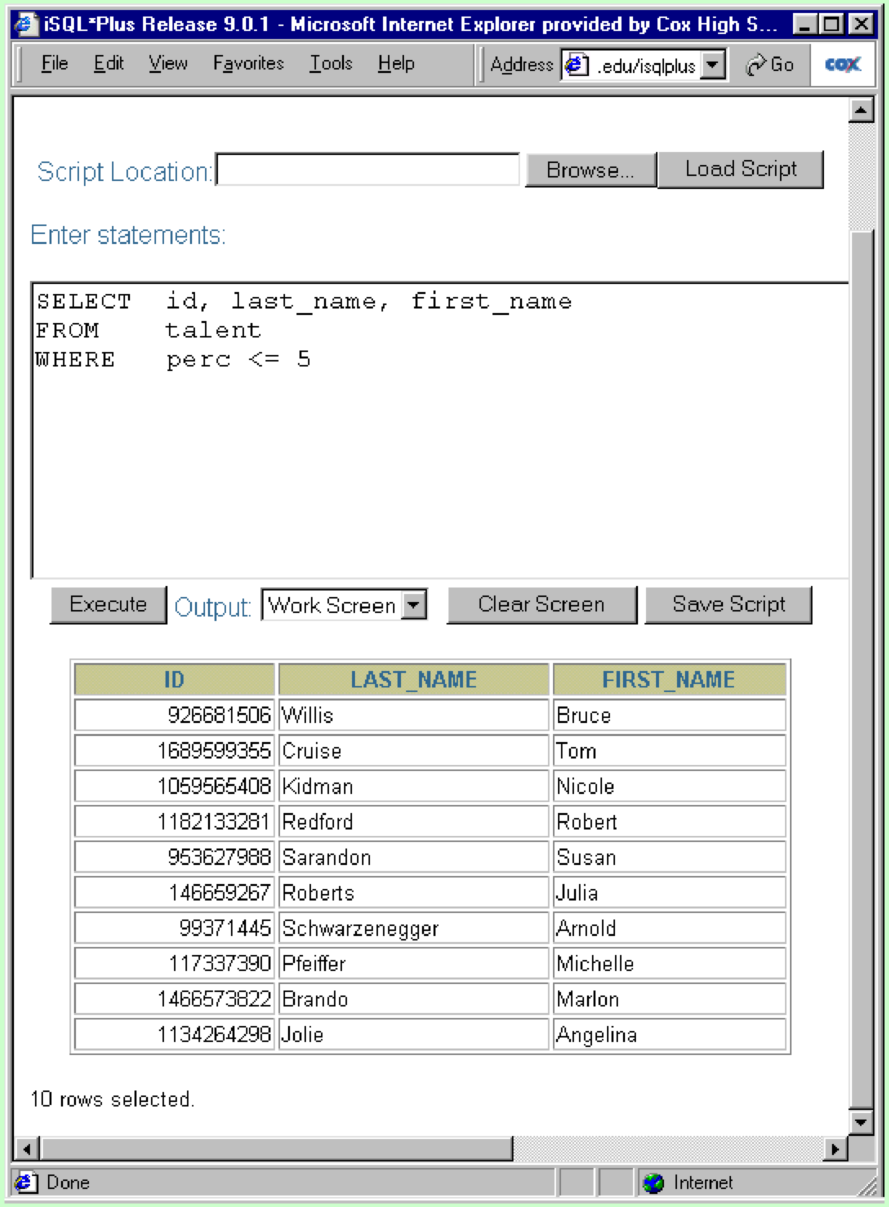
Step 1: Build the Table Build Chart (TBC)
Step 2: Double check your TBC solution
Step 3: Transform the TBC into code.



Column Name/Expression	id	last_name	first_name	perc
Table Name	talent	talent	talent	...
Alias				...
Criteria				<= 5
Display				...

```
SELECT  id, last_name, first_name
FROM     talent
WHERE   perc <= 5;
```

Module 03: WHERE Clause Page C-25: Problem 4-6 Solution



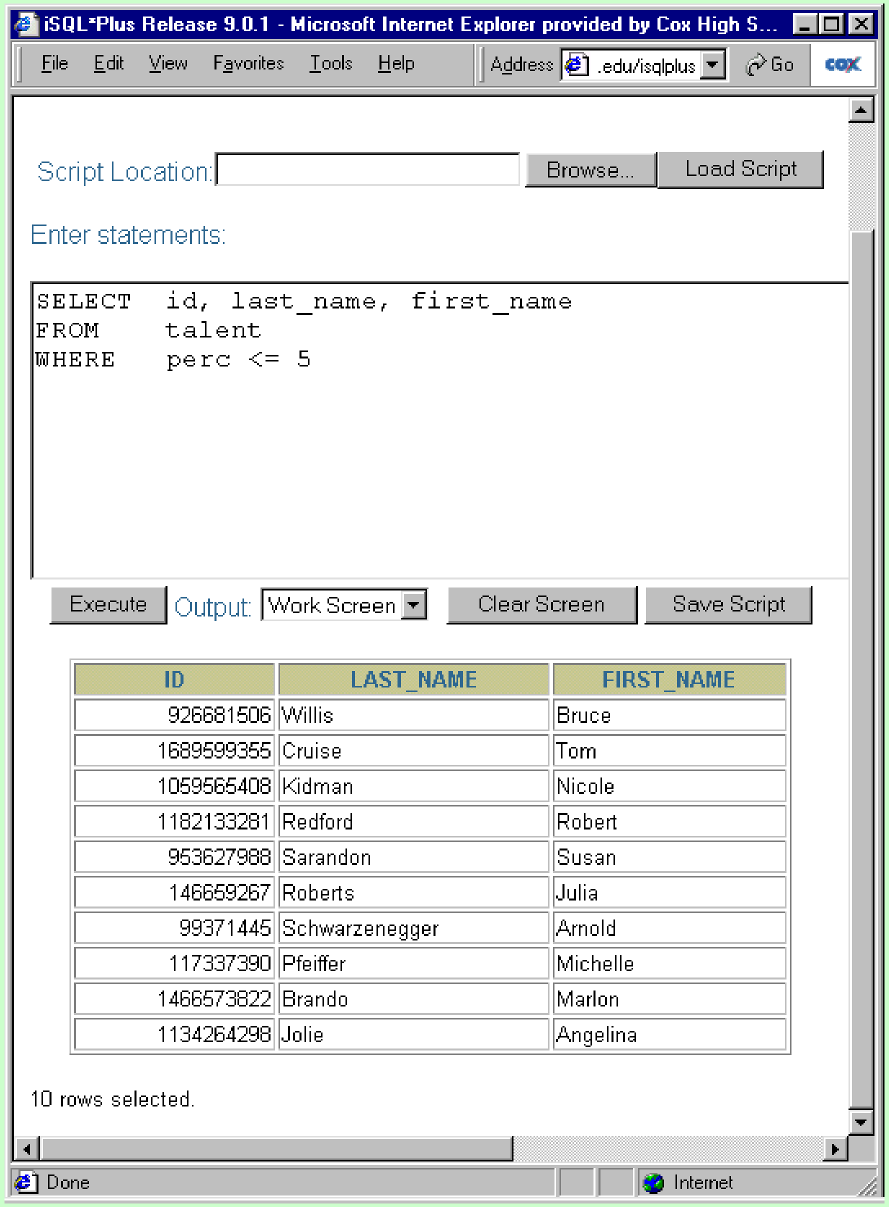
```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc <= 5
```

ID	LAST_NAME	FIRST_NAME
926681506	Willis	Bruce
1689599355	Cruise	Tom
1059565408	Kidman	Nicole
1182133281	Redford	Robert
953627988	Sarandon	Susan
146659267	Roberts	Julia
99371445	Schwarzenegger	Arnold
117337390	Pfeiffer	Michelle
1466573822	Brando	Marlon
1134264298	Jolie	Angelina

10 rows selected.

Here’s the solution.

Module 03: WHERE Clause Page C-26: Problem 4-6 Commentary



```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc <= 5
```

ID	LAST_NAME	FIRST_NAME
926681506	Willis	Bruce
1689599355	Cruise	Tom
1059565408	Kidman	Nicole
1182133281	Redford	Robert
953627988	Sarandon	Susan
146659267	Roberts	Julia
99371445	Schwarzenegger	Arnold
117337390	Pfeiffer	Michelle
1466573822	Brando	Marlon
1134264298	Jolie	Angelina

10 rows selected.

In this problem, the predicate expression applies a 'less than or equal to' comparison.

The symbol, or operator, for the less than or equal to condition is <=.

Page C-27: Problem 4-7 Statement

The user community asks for a report showing which of our clients are charged 10% or more.

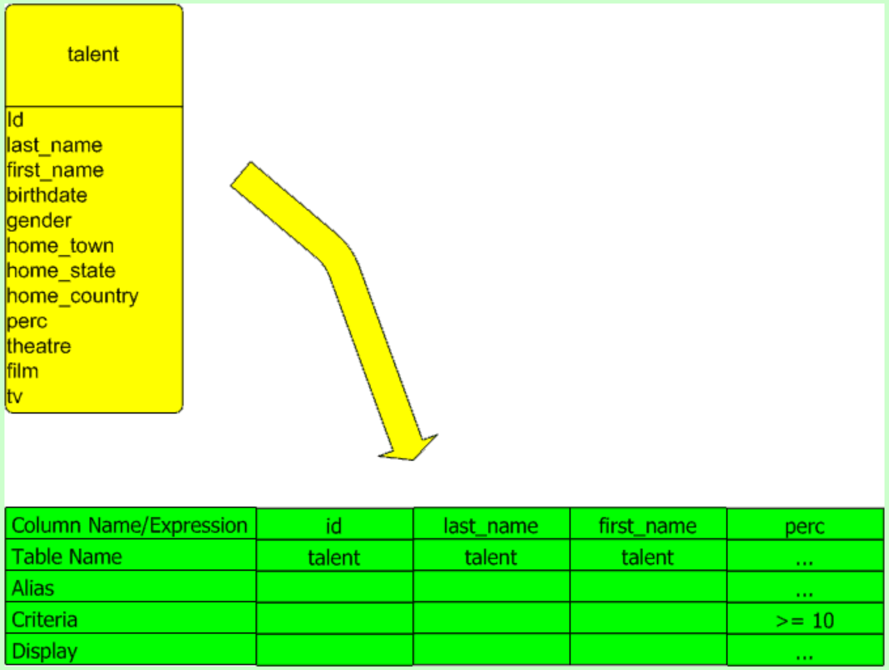
User need rephrased

Prepare a report showing id and name information for all of our clients whose percentage field is greater than, or equal to 10.

Page C-28: Problem 4-7 Design & Code

You won't notice much of a change in either our methodology, or our solution. The only item that will change will be the comparison operator.

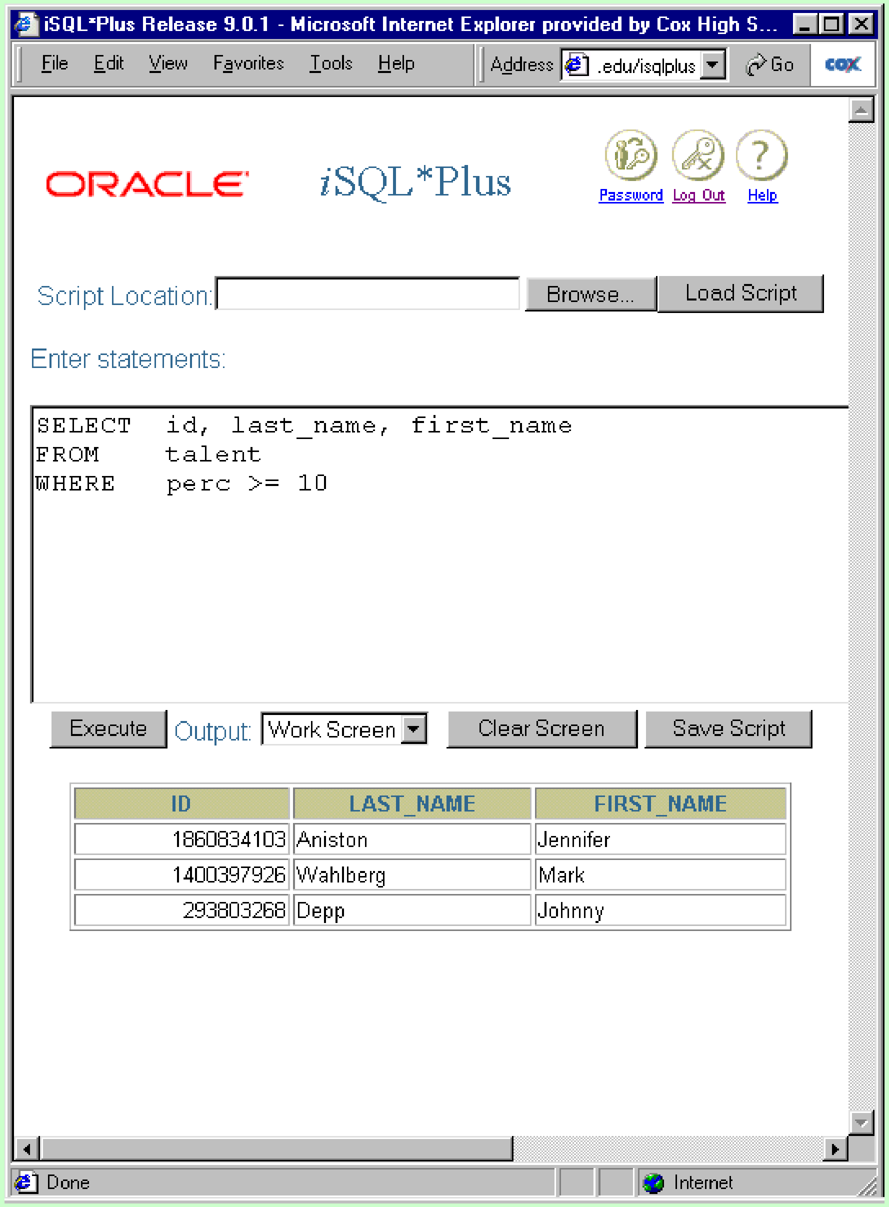
Step 1: Build the Table Build Chart (TBC)
Step 2: Double check your TBC solution
Step 3: Transform the TBC into code.



Column Name/Expression	id	last_name	first_name	perc
Table Name	talent	talent	talent	...
Alias				...
Criteria				>= 10
Display				...

```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc >= 10;
```

Module 03: WHERE Clause Page C-29: Problem 4-7 Solution

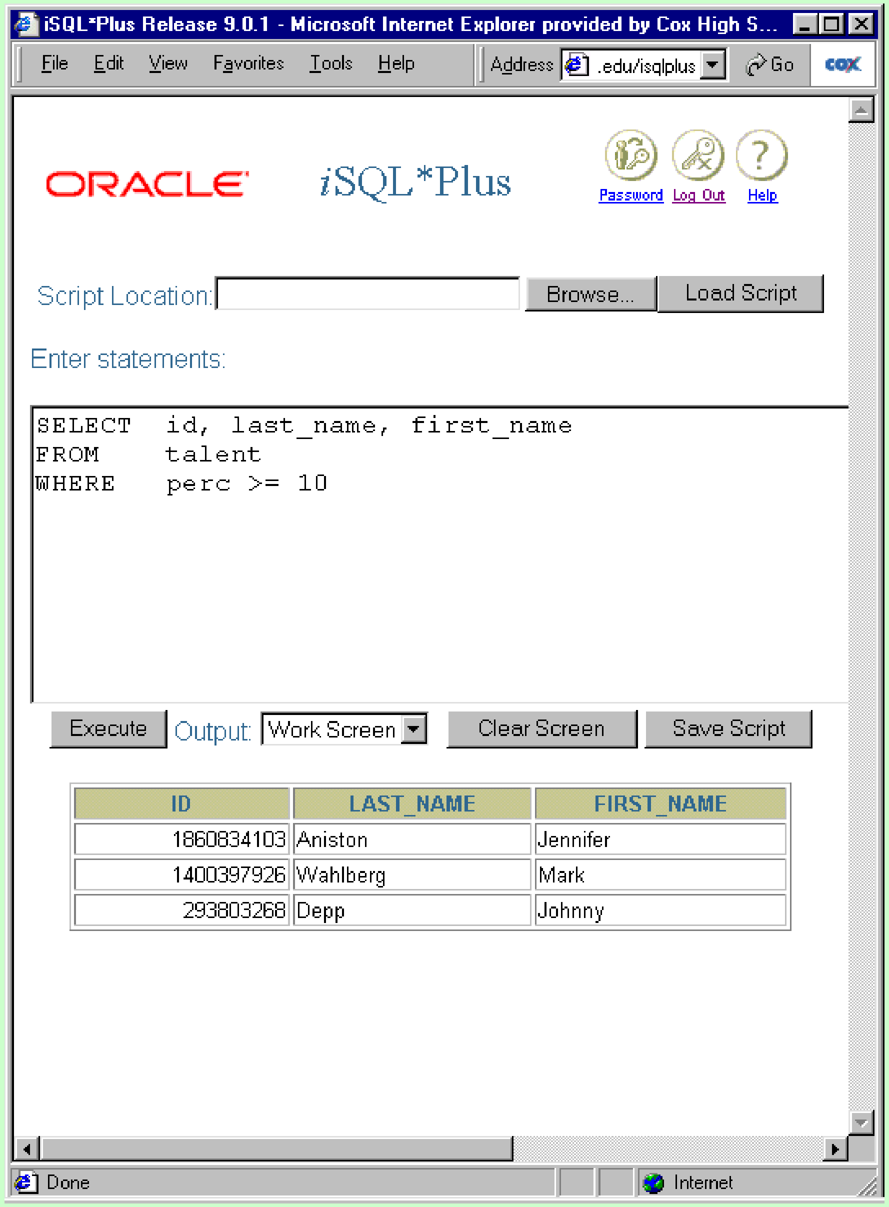


```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc >= 10
```

ID	LAST_NAME	FIRST_NAME
1860834103	Aniston	Jennifer
1400397926	Wahlberg	Mark
293803268	Depp	Johnny

Here’s the solution.

Page C-30: Problem 4-7 Commentary



```
SELECT  id, last_name, first_name
FROM    talent
WHERE   perc >= 10
```

ID	LAST_NAME	FIRST_NAME
1860834103	Aniston	Jennifer
1400397926	Wahlberg	Mark
293803268	Depp	Johnny

In this problem, the predicate expression applies a ‘greater than or equal to’ comparison.

The symbol, or operator, for the greater than or equal to condition is >=.

Module 03: WHERE Clause Page C-31: Reminder

Remember that it’s a good idea to type these programs in as you encounter them. The practice will help you learn the material more quickly.

Page C-32: Comparison Operations

We have just examined six of the most common comparison operations that are performed in any programming language.

These operations are used so frequently that special operators, or symbols, have been devised to simplify the task of coding these conditions.

Operation	Operator
Equality	=
Inequality	< >
Less than	<
Less than or equal to	< =
Greater than	>
Greater than or equal to	> =

Page D-1: Literals

Did you notice in that last series of examples that there were some occasions when we used single quote marks (‘) in the predicate expression?

I like quote marks. They make things very ‘explicit’. But enough about me, …

In SQL we use quote marks to identify string literals. Here’s another tech term.

What’s a string literal?

What’s a string? A string is a string of characters (letters, digits, or some combination thereof).

What’s a literal? A literal is something that should be interpreted literally by the computer. Recall from a previous example that ‘US’ is not literally the same as ‘USA’.

Data types

In the 1st semester SQL class we concentrate our studies on only three broad categories of data types.

More tech speak.

What's a data type? Obviously it's a type of data ☺

More precisely, a data type is a specification that defines what type of data is allowed in a column of data, and how that data will be stored internally. Every column in a relational database system includes a data type specification.

The three data type categories that we'll be studying are:

NUMERIC
CHARACTER
DATE

Page D-3 Numeric data types

Numeric data types are used for data that are numeric in nature.

Numeric columns hold numbers (numeric data) that might be manipulated mathematically. Take for example the two numeric columns: HOURLY_RATE and HOURS_WORKED. We might multiply these two columns to calculate the GROSS_PAY for an employee.

If we want to do ‘math’ stuff to any of our data, then the data needs to be defined by a data type that can support that math behavior.

Examples of numeric data include:

AGE, GRADE_POINT_AVERAGE, HEIGHT, WEIGHT, PRICE, ….

Page D-4: Character data types

Character data types are used to store data that is, by its nature, character in type.

Generally, character columns hold data values that include letters.

Examples of character data include:

NAME, ADDRESS, CITY, STATE, COURSE_TITLE, EYE_COLOR, HAIR_COLOR, MOVIE_TITLE, GENRE, ...

Character data is also referred to as alphanumeric data.

Page D-5: Date data types

The final category of data types that we'll consider is the date data type. Date columns store information about date and time data.

Date data types know how to behave as dates should behave.

The day after January 31st is February 1st, not January 32nd.

12:50 + 20 minutes = 1:10, not 12:70.

Examples of date data include:
BIRTH_DATE, DEATH_DATE,
PURCHASE_DATE,
DATE_AND_TIME_OF_PURCHASE,
...

Page D-6: Literals, revisited

Numeric literals – literally numbers, don't need any special treatment in your SQL programs. Just write them as numbers, and SQL can handle it.

```
WHERE  perc >= 8
```

Character literals – literally characters, are always enclosed within single quotes.

```
WHERE home_country <> 'USA'
```

Date literals – literally a date value (or time), are also enclosed within single quote marks, and must be written in the appropriate format. In an Oracle database system (that's what we're using) dates are written in DD-Mon-YYYY format. For example:

```
'10-JAN-2003'
```

Sometime DD-Mon-YY is permitted

```
'10-JAN-03'
```

Double quotes are only used for column aliases. Single quotes are used for all literal values, and pretty much everywhere else.

Page E-1: Character Comparisons

Each of the comparison operators may be used with character data.

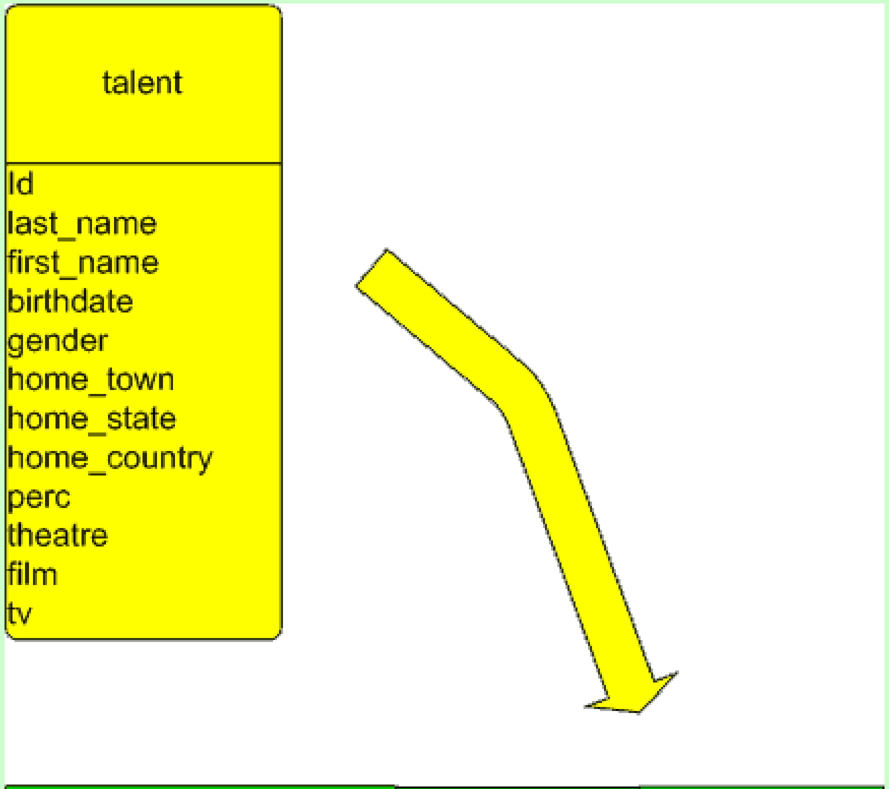
Here are some examples that deal with character data comparisons.

Module 03: WHERE Clause Page E-2: Problem 4-8 Statement

The user community needs a report showing all available information about Julia Roberts.

Module 03: WHERE Clause   Page E-3: Problem 4-8 Design & Code

Step 1: Build the Table Build Chart (TBC)
Step 2: Double check your TBC solution
Step 3: Transform the TBC into code.

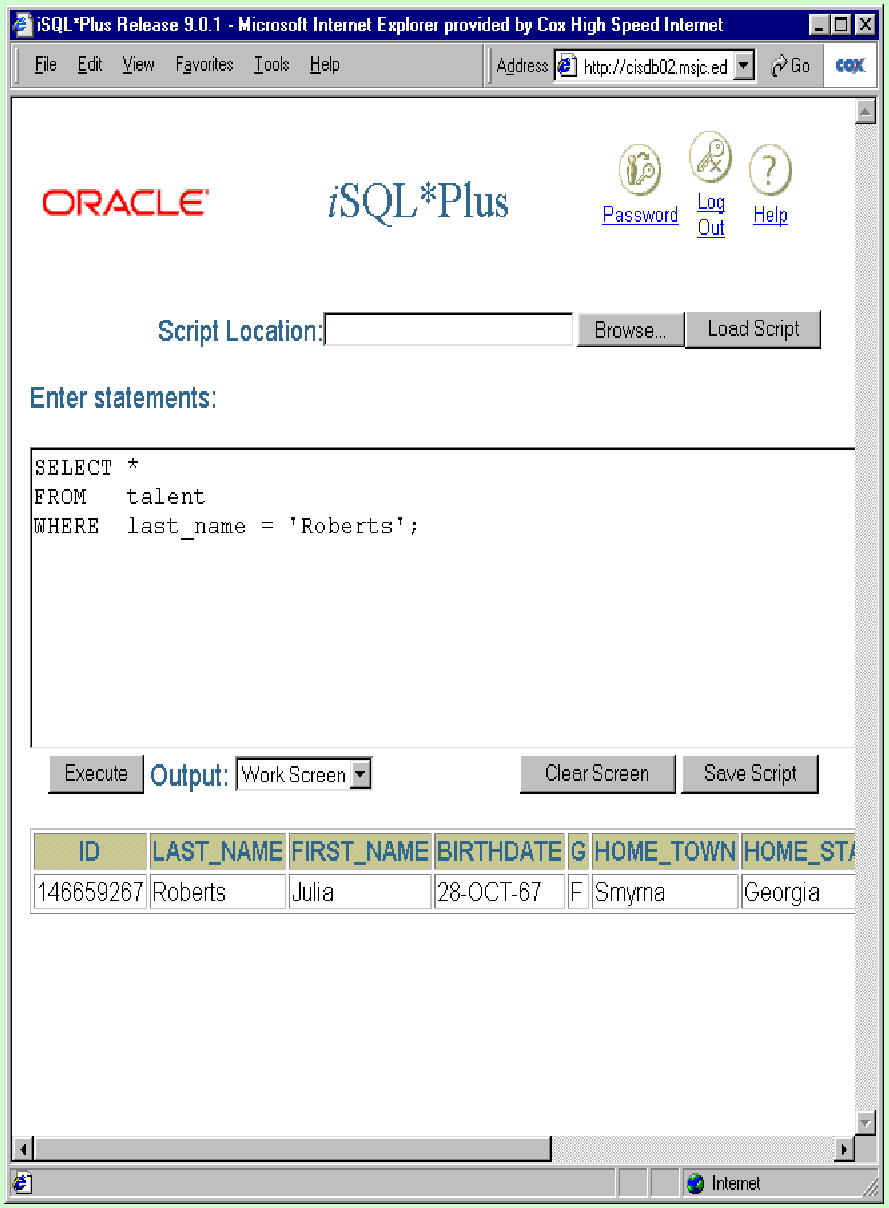


Column Name/Expression	id	last_name
Table Name	talent	talent
Alias		
Criteria		= 'Roberts'
Display		

```
SELECT  id, last_name, first_name
FROM    talent
WHERE   last_name = 'Roberts';
```

Module 03: WHERE Clause

Page E-4: Problem 4-8 Analysis


iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet
File Edit View Favorites Tools Help
Address http://cisdb02.msjc.ed
Go
ORACLE
iSQL*Plus
Password
Log Out
Help
Script Location:
Browse...
Load Script
Enter statements:
Execute
Output: Work Screen
Clear Screen
Save Script
Internet

```
SELECT *
FROM   talent
WHERE  last_name = 'Roberts';
```

ID	LAST_NAME	FIRST_NAME	BIRTHDATE	G	HOME_TOWN	HOME_STA
146659267	Roberts	Julia	28-OCT-67	F	Smyrna	Georgia

The character literal ‘Roberts’ is placed within single quotes.

This program answers the user’s question, but what would have been the result if there were more than one ‘Roberts’ in the database?

Page E-5: Problem 4-9 Statement

The user community needs a report showing all of our clients who were born prior to 1960.

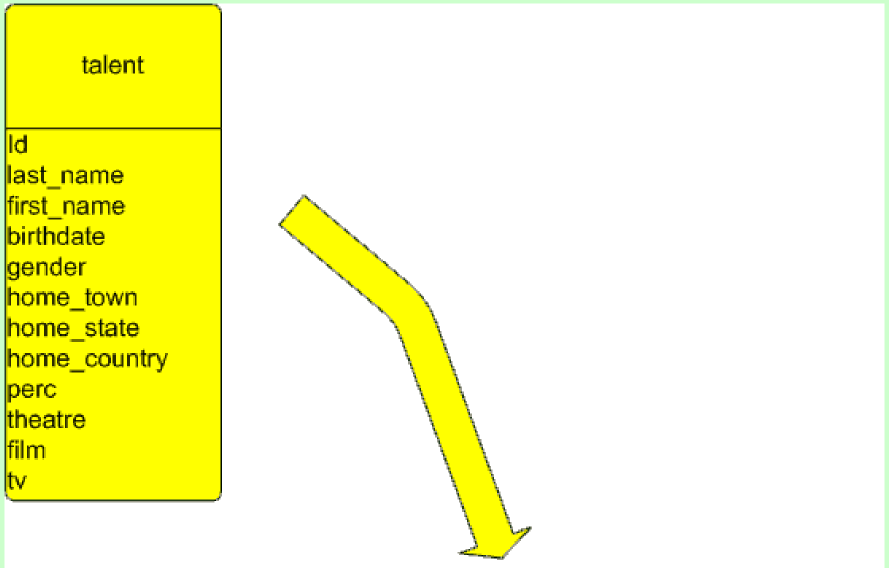
Rephrase: Show the name information and birth date for all of our clients who were born before 1960.

Module 03: WHERE Clause Page E-6: Problem 4-9 Design & Code

Step 1: Build the Table Build Chart (TBC)

Step 2: Double check your TBC solution

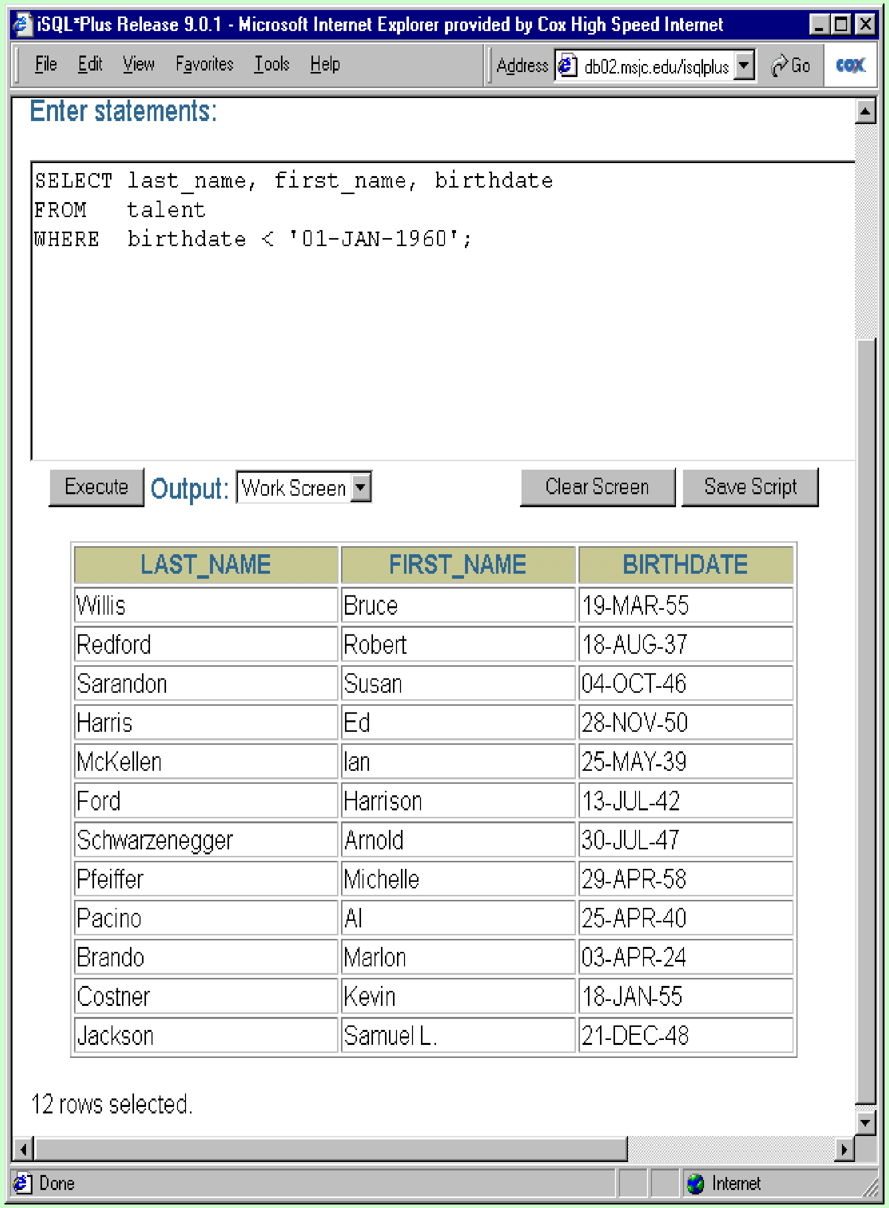
Step 3: Transform the TBC into code.



Column Name/Expression	last_name	first_name	birthdate
Table Name	talent	talent	...
Alias			...
Criteria			< '01-JAN-1960'
Display			...

```
SELECT   last_name, first_name, birthdate
FROM     talent
WHERE    birthdate < '01-JAN-1960';
```

Page E-7: Problem 4-9 Analysis


iSQL*Plus Release 9.0.1 - Microsoft Internet Explorer provided by Cox High Speed Internet
File Edit View Favorites Tools Help
Address db02.msjc.edu/isqlplus
Go
Enter statements:

```
SELECT last_name, first_name, birthdate
FROM   talent
WHERE  birthdate < '01-JAN-1960';
```

Execute
Output: Work Screen
Clear Screen
Save Script

LAST_NAME	FIRST_NAME	BIRTHDATE
Willis	Bruce	19-MAR-55
Redford	Robert	18-AUG-37
Sarandon	Susan	04-OCT-46
Harris	Ed	28-NOV-50
McKellen	Ian	25-MAY-39
Ford	Harrison	13-JUL-42
Schwarzenegger	Arnold	30-JUL-47
Pfeiffer	Michelle	29-APR-58
Pacino	Al	25-APR-40
Brando	Marlon	03-APR-24
Costner	Kevin	18-JAN-55
Jackson	Samuel L.	21-DEC-48

12 rows selected.

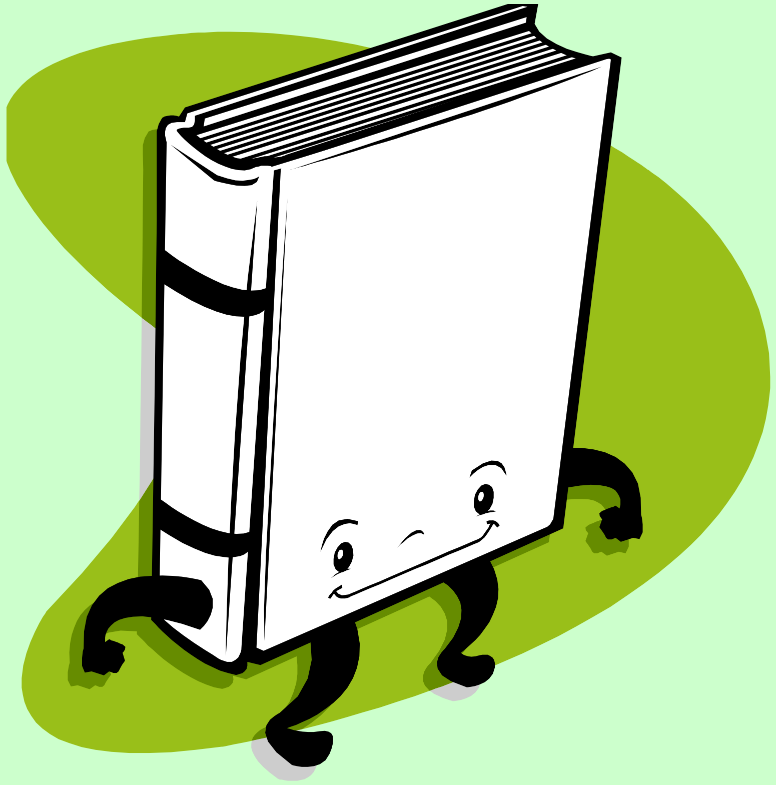
The date value is enclosed with single quotes, and is depicted in the proper format.

In Oracle, the format for date values is either:

	DD-MON-YYYY
Or	DD-MON-YY

Module 03: WHERE Clause

Page T-1: Terminology



WHERE clause

NULLs
TRUE, FALSE, UNKNOWN

Predicate expression
Comparison operation
Comparison operator

Equality, inequality
Less than, less than or equal to
Greater than, greater than or equal to

String, character, alphanumeric
Numeric data type
Date date type

Literal
String literal, character literal, numeric literal

Date format, DD-MON-YY, DD-MON-YYYY

Page Z-1: End Notes

Please drop me an email if you noticed any errors in this module. I'd also appreciate reading your comments, criticisms, and or suggestions as to how this module could be improved.

Thanks,

bil

That's All