University of Keele
School of Computing and Mathematics

Introduction to Oracle RDBMS and SQL*Plus

Oracle Corporation's *Relational* Database Management System (RDBMS) is one of the leading commercial database management systems designed for storing and handling large volumes of data and complex data retrieval/manipulation transactions. Oracle RDBMS has both server-side programs and applications (used by database administrators for configuring and management of the database) and client-side tools and utilities for developing end-user applications as well as using SQL commands for interacting with the database directly. Oracle RDBMS is a popular choice for financial databases as well as engineering and business intelligence solutions. For further details on Oracle's RDBMS features, architecture and development tools please visit http://www.oracle.com/index.html

We will use Oracle RDBMS's native query language, SQL (modern pronunciation: 'ess-cue-ell'; originally pronounced as Sequel: Structured English Query Language or simply Structured Query Language). SQL includes both data definition (DDL) and data manipulation (DML) commands or operators.

These exercises will focus on using the SQL*Plus client-side tool only, which is used to create and test command-line SQL queries. Our current Oracle build is version 11g.

In brief, …

- SQL statements consist of *reserved words* and *user-defined words*.
    - *reserved words* are a fixed part of SQL and must be used as specified
    - *user-defined words* represent names of various database objects, such as tables, columns and views.
- SQL statements are not case sensitive but literal characters (constants or values of attributes) are case sensitive.
- all non-numeric literals must be enclosed in single quotes, e.g. 'London'; numeric literals should NOT be enclosed in quotes, e.g. 650.00
- SQL statements are more readable if they are well-formatted. Each clause should preferably begin on a new line and the beginning of a clause should be lined up with the start of other clauses.

Entity ITEM

ITEM
ItemNumber (number) ItemName (string) Price (number) Colour (string) Weight (number) City (string)

Any entity can be implemented as a table in the relational model and relational DBMS. [Detailed study of the relational model can be found in the recommended literature]. In this part (Part 1), you will work with just one entity/table to ensure that you are familiar with the 'table' concept and some data manipulations commands.

Data definition/Creating Tables:

To create the table ITEM in Oracle, you need to activate a Windows-based SQL*Plus terminal from the Windows start menu or use a shortcut on the desktop (or via MS DOS command line: **cmd** > **sqlplus**).

You will be asked to enter your **Oracle user-id** and **password** (Your Keele user-id is currently set as both your Oracle user-id and password. You can change it by entering the '`password`' command at the SQL command prompt). Do not supply any parameters for the host string.

By default, SQL*Plus assumes that you are working on your 'home' or S:\ drive (If not, you can change the environment variables accordingly). It is a good idea to create a separate working directory for your sql scripts.

Use CREATE TABLE statement to define

- the name of the table
- types of columns
- primary key (row identifier) and foreign key(s) if required (ignore these for now)
- any other data access constraints (ignore these for now)

NUMBER: The NUMBER datatype stores fixed and floating-point numbers. Numbers of virtually any magnitude can be stored and are guaranteed portable among different systems operating Oracle Database, up to 38 digits of precision. **VARCHAR2 and VARCHAR**: The VARCHAR2 datatype stores variable-length character strings. When you create a table with a VARCHAR2 column, you specify a maximum string length (in bytes or characters) between 1 and 4000 bytes for the VARCHAR2 column. For each row, Oracle Database stores each value in the column as a variable-length field unless a value exceeds the column's maximum length, in which case Oracle Database returns an error. The VARCHAR datatype is synonymous with the VARCHAR2 datatype but do not use the VARCHAR data type because it is scheduled to be redefined for other uses.
Oracle built-in data types (more details and other data types and commands are available on *docs.oracle.com*)

```
CREATE TABLE ITEM (
     ItemNumber     VARCHAR2(5),
     ItemName  VARCHAR2(20),
     Price          NUMBER,
     Colour         VARCHAR2(8),
      Weight        NUMBER,
     City        VARCHAR2(15), PRIMARY KEY (ItemNumber));
```

++ The `PRIMARY KEY` clause enforces entity integrity (see reference literature). `ItemNumber` is a unique identifier for ITEM tuples or records.

++ SAVE this script as **itemdefn.sql** (see SQL tips on final page)

To remove a table from your table space use the command:

DROP TABLE ‘tablename’ - replacing ‘tablename’ with the name of the table that you want to drop.

Data manipulation operations:

- INSERT: inserts new tuples/rows in an existing table
- SELECT: retrieves tuples/rows from an existing table
- UPDATE: updates data in tuples/rows in an existing table.
- DELETE: deletes tuples/rows from an existing table

NB: The term 'update' in the generic sense refers to three commands: INSERT, DELETE and UPDATE. The commands are also referred to as 'update operations’.

Inserting tuples:

```
INSERT INTO ITEM VALUES
  ('T1', 'Apple', 15, 'Green', 100, 'London');

INSERT INTO ITEM VALUES
  ('T2', 'Calculator', 50, 'Blue', 25, 'Sydney');
```

* NOW INSERT AT LEAST 30 DIFFERENT ITEMS WITH SUITABLE VALUES *

++ SAVE these commands in a script file, e.g., as **itemdata.sql**

Examples of retrieval queries:

[Save these queries as *q*1, *q*2, *q3*, and so on]

Get an item number for each item

```
SELECT ItemNumber
     FROM ITEM;
```

Get all details for each item

```
SELECT ItemNumber, ItemName, Price, Colour, Weight, City FROM
ITEM;
```

This query could simply be written as

```
SELECT   *
     FROM ITEM;
```

Get the item numbers for all items located in Paris

```
SELECT ItemNumber
     FROM ITEM WHERE CITY = 'Paris';
```

Get the total number of items

```
SELECT COUNT(*)
     FROM ITEM;
```

Get item numbers and prices for all apples located in London and with a price of over 20 units

```
SELECT ItemNumber, Price
     FROM ITEM
          WHERE ItemName = 'Apple' AND CITY = 'London' AND Price
          > 20;
```

Get item numbers and prices for all apples located in London and with a price of over 20 units, in descending order of the price.

```
SELECT ItemNumber, Price
     FROM ITEM
          WHERE ItemName = 'Apple' AND CITY = 'London' AND Price
          > 20
          ORDER BY Price DESC;
```

(Default is ASC)

Ordering on multiple columns

```
SELECT ItemNumber, Price
     FROM ITEM
          WHERE CITY = 'London'
          ORDER BY ItemNumber, Price DESC;
```

Updating data:

Change the price of item T2 to 15 and increase its weight by 5

```
     UPDATE ITEM
          SET PRICE = 15,
         WEIGHT = WEIGHT +  5
          WHERE ItemNumber = 'T2';
```

Deleting data:

Delete item T1

```
DELETE FROM ITEM
WHERE ItemNumber = 'T1';
```

Delete all items

```
DELETE
FROM ITEM;
```

You can re-load the data by executing your **@itemdata.sql** (see 'helpful commands' section for more help)

```
/end PART 1
```

Some helpful commands on using sqlplus:

1. If an SQL command fails, type
   ed
   This will take you to the editor (default Notepad) so that you can edit your current sql command which is in the buffer.
   When you have finished editing and exited from the editor type
   run
   or, simply
   r
   to run the current sql command.

2. To save a command in a file type
   save <filename>
   This will save the current command (which is in a buffer) in the specified file which will have the extension .sql
   To edit this file from within sqlplus type
   ed <filename>

From within sqlplus you do not have to give the .sql extension.
To run this file, type

start <filename>

or

@<filename>

Again without the .sql extension.
(The sqlplus command run described in 1. above runs the command currently in the buffer. run does not recognise a filename.)

3. To check what tables you have access to type

   select * from tab;

4. To list or describe the attributes of a table type

   descr supp

   (no semicolon needed).

5. To display a system variable, such as sysdate and user, type

   SELECT sysdate, user
   FROM dual;

6. To stop the output scrolling wither use the scrollbar or type

   SET pause on
   SET pagesize 23

   There are many other parameters that you can set with SET.

7. To send your output to a file type

   SPOOL report

   To stop the output going to the file type

   spool off

   To print the file, exit from sql and type

   lpr report.lst

8. To undo all the sql commands (update, insert etc) that you have executed from the beginning of the sqlplus session, type

   rollback;

9. To commit all the commands (update, insert etc) that you have executed from the beginning of the sqlplus session, type
   commit;

   Exiting from sqlplus has the same effect.