School of Computing and Information Technology
University of Wollongong

Session: **Summer 2019**
Lecturer: **Janusz R. Getta**

CSCI835 Database Systems (Bridging Subject)
Practice
22 July 2019

Task 1
`CREATE TABLE` statement

Consider the following collection of relational schemas together with the domain constraints.

`EMPLOYEE(enum, first-name, last-name, salary)`
primary key = (`enum`)
domain constraints:
- `enum` is a positive number 7 digits long,
- `first-name`, `last-name` are variable size strings no longer than 30 characters,
- `salary` is a positive number with 6 digits before decimal do and 2 digits after decimal dot,
- the values in all columns are compulsory.

`DRIVER(enum, license, category, experience)`
primary key = (`license`)
candidate key = (`enum`)
foreign key = (`enum`) references `EMPLOYEE(enum)`
domain constraints:
- `license` is a fixed size string of 7 characters,
- `category` is a single character, only the following values are possible: A, B, C,
- `experience` is a variable size strong of maxim 2000 characters,
- the values in all columns are compulsory.

No automatic deletion of the rows is allowed when a referenced row with a respected value of primary key is deleted from the other table.

`ADMIN(enum, position)`
primary key = (`enum`)
foreign key = (`enum`) references `EMPLOYEE(enum)`
domain constraints:
- `position` is a variable size string no longer than 30 characters.
- the values in all columns are compulsory.

Automatic deletion of the rows is required when a referenced row with a respected value of primary key is deleted from the other table.

`TRUCK(rego, capacity, manufacturer, model)`

primary key = (rego)
domain constraints:

rego is a fixed size string o 8 characters
capacity is a floating point number
manufacturer and model are variables size strings no longer than 50 characters
the values in the columns manufacturer and model are optional.

TRIP(origin, destination, license, rego, tdate)
primary key = (license, rego, tdate)
foreign key1 = (license) references DRIVER(license)
foreign key2 = (rego) references TRUCK(rego)
domain constraints:

origin and destination are variable size strings no longer than 30 characters,
tdate is of type DATE.

No automatic deletion of the rows is allowed when a referenced row with a respected value of primary key is deleted from the other table.

Your task is to implement SQL script solution1.sql with CREATE TABLE statements that can be used to implement the relational schemas.

When a script file solution2.sql is ready connect to Oracle either through command line interface SQLPlus or graphical user interface SQL Developer and process your script file.

To list the names of relational tables created you can use a script list.sql.

If processing of the file returns the errors then you must eliminate the errors! Processing of your script must return NO ERRORS !

It is recommended to create a script drop.sql that drops all relational tables created by processing of a script solution1.sql and it is recommend to execute drop.sql after each processing of solution1.sql

---

Task 2
`ALTER TABLE` statement

SQL script `dbcreate2.sql` contains `CREATE TABLE` statements that can be used to create a small relational that contains information about employees, department and locations of department (it is the same relational database as it is frequently used in Cookbook). A conceptual schema of the database is included in a file `schema2.pdf`.

Analyse, the contents of a script `dbcreate2.sql` to find out how the relational tables are implemented. Connect to Oracle database server either through command line interface SQLPlus or graphical user interface SQL Developer. process a script `dbcreate2.sql` to create the relational tables.

Your task is to implement SQL script `solution2.sql` with only (!) `ALTER TABLE` statements that implement the following modifications of the original relational tables included in the sample database.

(1) We would like to be able to add information about the total number of employees working at a particular location. Assume that no more than 1000 employees can work at the same location.

(2) We would like to increase the maximum length of department name to 50 characters;

(3) We would like to replace information about chairman of each department with information about manager of each department assuming that one of the employees is a manager of a department.

(4) We would like to increase maximum budget of a department to 5000.

(5) We would like to allow for automatic deletion of a row from `EMPLOYEE` table when a row from `DEPARTMENT` table is deleted. Additionally, when a row from a relational table `DEPARTMENT` is deleted then the respective values of foreign keys in a relational table `DEPTLOC` must be set to `NULL`.

It is recommended to use a script `dbdrop2.sql` to drop all relational tables modified during the processing of a script `solution2.sql` and then to re-create the original database with a script `dbcreate2.sql`. In such a way your script always operates on the original structures of the sample database.

Task 3
Simple data manipulation statements

Download the files `dbcreate.sql` and `dbdrop.sql`.

Connect to Oracle database server either through command line interface SQLPlus or graphical user interface SQL Developer.

To create and to load the relational tables of a sample database, process SQL script `dbcreate.sql`. A script `dbdrop.sql` can be used to drop the relational tables. Do not drop the relational tables now.

Your task is to use `INSERT` and/or `DELETE` and/or `UPDATE` statements of SQL to implement a script file `solution3.sql` that performs the data manipulations listed below. You must drop all tables of the sample database, re-create all tables of the database and load data into the database with `dbcreate.sql` script before implementation of this task.

Your SQL statements must operate on the sample database loaded with data.

An important condition is that you must only use `INSERT` and/or `DELETE` and/or `UPDATE` statements of SQL. No other statements of SQL are allowed. It means that you are not allowed to change any consistency constraints imposed on the contents of the data base like, for example suspension of foreign key constraints, etc.

Note, that implementation of the actions listed below may require more than one SQL statement.

(1) Insert into the database information about a trip performed by a new truck driver. A new driver obtains employee number 21. His personal record is the following.

Harry F. Potter, born 21st , November, 1984, living in Waga Waga, Railway Street 25, NSW 2767, driver license number 666, and his present status is "on leave". The driver performed a trip from Sydney to Melbourne on 10th, April 2016. The driver used a truck with registration number PKR768.

The trip consisted of the following two legs: the first leg from Sydney to Canberra and then the second leg from Canberra to Melbourne.

(2) A driver with an employee number 7 decided to quit a job. Remove all information about the driver and about all trips performed by the driver.

(3) A registration number of a truck with the present registration number `SST005` has been changed to `PKR856`.

Task 4
`SELECT` statement: statements with `WHERE`, `GROUP BY` and `HAVING` clauses

Download the files `dbcreate.sql` and `dbdrop.sql`.

Connect to Oracle database server either through command line interface SQLPlus or graphical user interface SQL Developer.

To create and to load the relational tables of a sample database, process SQL script `dbcreate.sql`. A script `dbdrop.sql` can be used to drop the relational tables. Do not drop the relational tables now.

Your task is to implement the following queries as `SELECT` statements of SQL and save the statements in SQL script file `solution4.sql`.

(1) Find full names of employees living in states NSW or WA.

(2) Find full information about trucks that are not available just now.

(3) Find dates of all trips performed by a driver with license number 10001 who used a truck different from a truck with registration PKR768.

(4) List all information about the trips performed by the drivers with license numbers 10001, 10002, and 10003. List the result in the descending order of license numbers and for all trips with the same license number in the ascending orders of truck registration numbers.

(5) List full names of all employees in uppercase format in the descending order of last names.

(6) Find the driver license numbers (attribute `LNUM`) of all drivers who performed at least one trip in 2015 or at least one trip in 2016.

(7) Find the driver license numbers (attribute `LNUM`) of all drivers who performed at least one trip in 2015 and at least one trip in 2016.

(8) Find the driver license numbers (attribute `LNUM`) of all drivers together with the total number of trips performed by each driver. You may ignore the drivers who performed no trips so far. For each driver list a driver license number to together with the total number of trips in one line, then next driver license number with the total number of trips in the next line, and so on.

(9) Find the driver license numbers (attribute `LNUM`) of all drivers who performed more than 3 trips.

(10) Find full names of all drivers whose present status is “on leave”.

Task 5
`SELECT` statement: joins, antijoins, `GROUP BY` and `HAVING` clauses

Download the files `dbcreate.sql` and `dbdrop.sql`.

Connect to Oracle database server either through command line interface SQLPlus or graphical user interface SQL Developer.

To create and to load the relational tables of a sample database, process SQL script `dbcreate.sql`. A script `dbdrop.sql` can be used to drop the relational tables. Do not drop the relational tables now.

Your task is to implement the following queries as `SELECT` statements of SQL, save and save the statements in SQL script file `solution5.sql` and process the script file.

The queries listed below must be implemented as `SELECT` statements with `JOIN` operation.

(1) Find full names of all drivers who used a truck with a registration PKR856 at least one time.

(2) Find the pairs of truck registration number (attribute `REGNUM`) and driver license number (attribute `LNUM`) such that the present status of a truck is different from the present status of a driver.

(3) Find the numbers of trips (attribute `TNUM`) of all trips that included 2 successive (adjacent) legs from Melbourne to Sydney and from Sydney back to Melbourne. This query must be implemented as a self-join query !

(4) Find the driver license numbers (attribute `LNUM`) of all drivers together with the registration numbers of all trucks used by the drivers. If a driver has not used any truck so far then his driver license number must be listed with `NULL`.

(5) Find the driver license numbers (attribute `LNUM`) of all drivers together with the total number of trips performed by each driver. If a driver performed no trips so far then his driver license number must be listed with 0. For each driver list a driver license number to together with the total number of trips in one line, then next driver license number with the total number of trips in the next line, and so on.

(6) List full information about the trucks whose registration number starts from a letter P and ends with a digit 8.

(7) Find the total number of times the trips either started or ended or passed through Perth.

(8) Find the largest capacity and the smallest weight of all trucks and list the results in one line.

(9) Find full names of all drivers whose date of birth is unknown.

(10) Find the registration numbers of all truck used for at least one trip in 2012. Do not list duplicated registration numbers.

(11) Find full names of all drivers whose present status is “on leave”. The query must be implemented as a nested query.

(12) Find full names of all drivers whose present status is “on leave”. The query must be implemented as a correlated nested query (query with an existential quantifier).

(13) Find full names of all drivers who used a truck with a registration PKR856 at least one time. The query must be implemented as a nested query.

(14) Find the driver license numbers (attribute `LNUM`) of all drivers who never used a truck with a registration PKR856. The query must be implemented as a nested query.

(15) Find the driver license numbers (attribute `LNUM`) of all drivers who never used a truck with a registration PKR856. The query must be implemented as a correlated nested query (a query with a negated existential quantifier).

Task 6
Advanced data manipulations
Download the files `dbcreate.sql` and `dbdrop.sql`.

Connect to Oracle database server either through command line interface SQLPlus or graphical user interface SQL Developer.

To create and to load the relational tables of a sample database, process SQL script `dbcreate.sql`. A script `dbdrop.sql` can be used to drop the relational tables. Do not drop the relational tables now.

Your task is to implement the following advanced manipulations on data in SQL, save your implementations in SQL script file `solution6.sql` and process the script.

(1) Create a relational table `EMPNAME` that consists of the columns `ENUM`, `FNAME`, `INITIALS`, and `LNAME` copied from a relational table `EMPLOYEE`. Enforce appropriate consistency constraints on a relational table `EMPNAME`.
(2) Create an empty relational table `TRIP2015`, which has the following columns: `TNUM`, `LNUM`, `REGNUM` with the same types as the columns with the respective names in a relational table `TRIP`. You can re-use slightly updated `CREATE TABLE` statement included in a script `dbcreate3.sql`. Enforce primary key and referential integrity constraints on a relational table `TRIP2015`. Next, copy information about all trips performed in 2015 from `TRIP` to `TRIP2015`.
(3) Use a single `UPDATE` statement to change a status of truck (attribute `STATUS`) to "maintained" for all trucks that have been used at least one time in 2005 and earlier.
(4) Add a column `TOTALTRIPS` to a relational table `DRIVER`. A type of the column must be `DECIMAL(3)`. Next, insert into a column `TOTALTRIPS` the total number of trips performed by each driver.
(5) Use a single `DELETE` statement to remove all legs of all trips performed in 2015.

End of specification