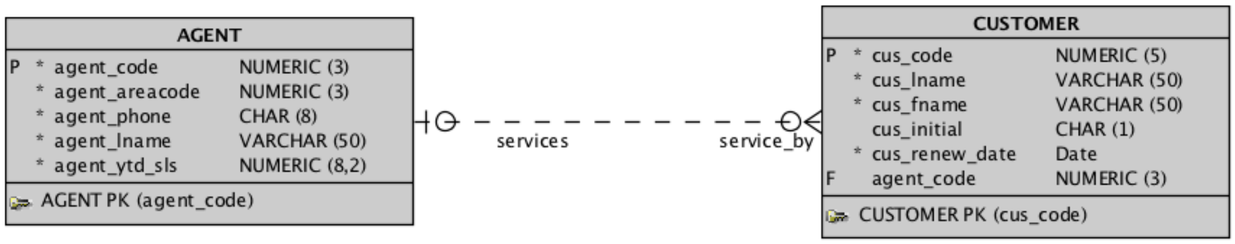
Week 3 Tutorial
SQL Data Definition Language (DDL) and Data Maintenance

When creating schema files, you should always also create a drop file or add the drop commands to the top of your schema file. You should drop the tables using the:

drop table tablename purge;

syntax. The drop table statements should list tables in the reverse order of your create table order so that FK relationships will be able to be removed successfully. Should a syntax error occur while testing your schema, you simply need to run the drop commands to remove any tables which may have been created.



The data model above represent figure 3.3 from Coronel & Morris. There are two different ways of coding this model as a set of create table statements.

Using table constraints

SQL constraints are classified as column or table constraints; depending on which item they are attached to:

```
create table agent
  (
    agent_code     number (3) constraint agent_pk primary key,
    agent_areacode number (3) not null ,
    agent_phone    char (8) not null ,
    agent_lname    varchar2 (50) not null ,
    agent_ytd_sls  number (8,2) not null
  ) ;
```

This is a declaration of the primary key as a column constraint

```
create table agent
  (
    agent_code     number (3) not null ,
    agent_areacode number (3) not null ,
    agent_phone    char (8) not null ,
    agent_lname    varchar2 (50) not null ,
    agent_ytd_sls  number (8,2) not null,
    constraint agent_pk primary key ( agent_code )
  ) ;
```

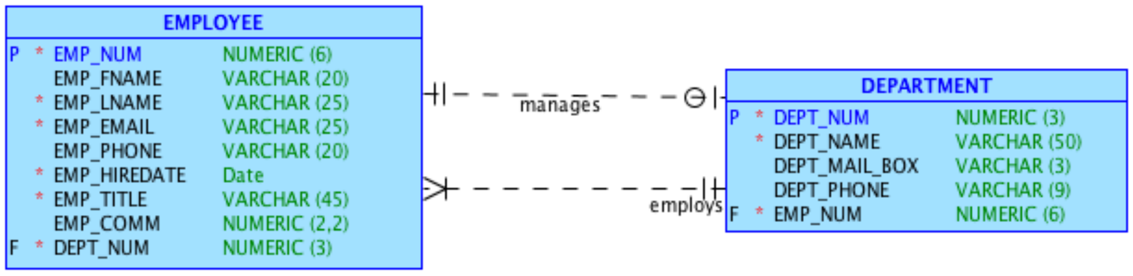
Here the primary key has been declared as a table constraint, at the end of the table after all column declarations have been completed. In some circumstances, for example a composite primary key you must use a table constraint since a column constraint refers only to a single column.

The create table statements for the two tables in fig 3-3 would be:

```
create table agent
  (
    agent_code     number (3) not null ,
    agent_areacode number (3) not null ,
    agent_phone    char (8) not null ,
    agent_lname    varchar2 (50) not null ,
    agent_ytd_sls  number (8,2) not null,
    constraint agent_pk primary key ( agent_code )
  ) ;


create table customer
  (
    cus_code       number (5) not null ,
    cus_lname      varchar2 (50) not null ,
    cus_fname      varchar2 (50) not null ,
    cus_initial    char (1) ,
    cus_renew_date date not null ,
    agent_code     number (3),
    constraint customer_pk primary key ( cus_code ),
    constraint customer_agent_fk foreign key ( agent_code)
       references agent ( agent_code ) on delete set null
  ) ;
```

In some circumstances this approach cannot be used. Can you see what the issue is with trying to create the two tables depicted below?



In such a situation an alternative approach to declaring constraints needs to be adopted.

Using ALTER table commands

In this approach the tables are declared without constraints and then the constraints are applied via the ALTER TABLE command (see section 7.5 of Coronel & Morris).

```
create table agent
  (
    agent_code     number (3) not null ,
    agent_areacode number (3) not null ,
    agent_phone    char (8) not null ,
    agent_lname    varchar2 (50) not null ,
    agent_ytd_sls  number (8,2) not null
  ) ;

alter table agent add constraint agent_pk primary key
   ( agent_code ) ;

create table customer
  (
    cus_code       number (5) not null ,
    cus_lname      varchar2 (50) not null ,
    cus_fname      varchar2 (50) not null ,
    cus_initial    char (1) ,
    cus_renew_date date not null ,
    agent_code     number (3)
  ) ;

alter table customer add constraint customer_pk primary key
    ( cus_code ) ;

alter table customer add constraint customer_agent_fk foreign key
   ( agent_code ) references agent ( agent_code )
   on delete set null;
```

After creating the tables we need to insert the data, for AGENT the insert will have the form:

```
insert into agent values (501,713,'228-1249','Alby',132735.75);
```

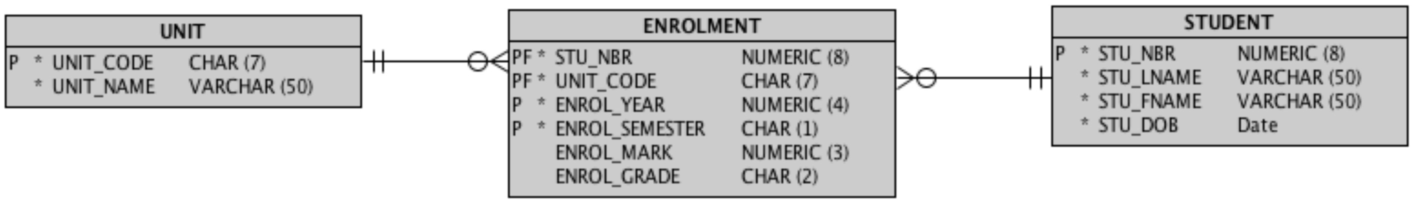
for customer:

```
insert into customer values(10010,'Ramas','Alfred','A',
    '05-Apr-2014',501);
```

It is important to note that for the insert into customer we are using the default Oracle date format of dd-mon-yyy – in the near future we will correct this and allow any date format via the oracle function *todate*.

Lab Tasks

Using the model from the DDL lecture for student, unit and enrolment:



Q1. Code a schema file to create these three tables, noting the following extra constraints:

- stu_nbr > 10000000
- unit_name is unique in the UNIT table
- enrol_semester can only contain the value of 1 or 2 or 3

In implementing these constraints you will need to make use of CHECK clauses (see Coronel & Morris section 7.2.6).

Ensure your script file has appropriate comments in the header, includes the required drop commands and includes echo on and echo off commands.

Run your script and create the three required tables. Save the output from this run.

Q2. Insert the following data into the tables specified using the SQL INSERT statement:

STUDENT:

stu_nbr	**stu_lname**	**stu_fname**	**stu_dob**
11111111	Bloggs	Fred	01-Jan-1990
11111112	Nice	Nick	10-Oct-1994
11111113	Wheat	Wendy	05-May-1990
11111114	Sheen	Cindy	25-Dec-1996

UNIT:

unit_code	**unit_name**
FIT5132	Introduction to Databases
FIT5016	Project
FIT5111	Student's Life
FIT9999	FIT Last Unit

ENROLMENT:

stu_nbr	unit_code	enrol_year	enrol_semester	enrol_mark	enrol_grade
11111111	FIT5132	2013	1	35	N
11111111	FIT5016	2013	1	61	C
11111111	FIT5132	2013	2	42	N
11111111	FIT5111	2013	2	76	D
11111111	FIT5132	2014	2		
11111112	FIT5132	2013	2	83	HD
11111112	FIT5111	2013	2	79	D
11111113	FIT5132	2014	2		
11111113	FIT5111	2014	2		
11111114	FIT5111	2014	2		

Ensure you make use of COMMIT to make your changes permanent. Check that your data has inserted correctly by using the SQL command SELECT * FROM *tablename* **and** by using the SQL GUI (select the table in the right hand list and then select the Data tab).

Q3. Update the name of FIT5016 to "IT Project", check that the name has been updated

Q4. Remove FIT9999 from the database

Q5. Add a new column to the UNIT table which will represent points for the unit - the default should be 6 points (hint use the ALTER command). After you have added the new column insert a test new unit. Check that the new insert has worked correctly.

Q6. A table can also be created based on an existing table, and immediately populated with contents by using a SELECT statement within the CREATE TABLE statement.

For example to create a table called FIT5132_STUDENT which contains the enrolment details of all students who have been or are currently enrolled in FIT5132, we would use:

```
create table FIT5132_STUDENT
 as select *
 from enrolment
 where unit_code = 'FIT5132';
```

Create this table, check it exists and the data it contains.

Q7. Create a sequence for the STUDENT table called STUDENT_SEQ that starts at 11111115 and increases by 1. Check that the sequence exists in two ways (using SQL and browsing your SQL Developer connection objects).

Q8. Add a new student (MICKEY MOUSE) using the student sequence – pick any STU_DOB you wish. Check that your insert worked.