**SQL Lab**

Download Chinook Database for Postgres SQL: <https://raw.githubusercontent.com/xivSolutions/ChinookDb_Pg_Modified/master/chinook_pg_serial_pk_proper_naming.sql>

## 2.0 SQL Queries

[X]In this section you will be performing various queries against the Oracle Chinook database.

## 2.1 SELECT

Task – Select all records from the Employee table.

**SELECT** \* **FROM** employee;

Task – Select all records from the Employee table where last name is King.

**SELECT** \* **FROM** employee **WHERE** last\_name = 'King';

Task – Select all records from the Employee table where first name is Andrew and REPORTSTO is NULL.

**SELECT** \* **FROM** employee **WHERE** first\_name = 'Andrew' **AND** reports\_to **IS** **NULL**;

## 2.2 ORDER BY

Task – Select all albums in Album table and sort result set in descending order by title.

**SELECT** \* **FROM** album **ORDER** **BY** title;

Task – Select first name from Customer and sort result set in ascending order by city

**SELECT** first\_name **FROM** customer **ORDER** **BY** city;

## 2.3 INSERT INTO

Task – Insert two new records into Genre table

**INSERT** **INTO**

genre(**name**)

**VALUES**

('Dubstep'),

('Drum and Bass');

Task – Insert two new records into Employee table

**INSERT** **INTO**

employee(last\_name, first\_name, title, reports\_to, birth\_date,

hire\_date, address, city, state, country, postal\_code,

phone, fax, email)

**VALUES**

('Smith', 'John', 'IT Staff', 6, '1976-02-18 00:00:00',

'1992-02-18 00:00:00', '123 Apple St', 'Miami', 'FL', 'USA', '33015',

'+1 (786) 123-4567', '+1 (786) 321-4567', 'email@gmail.com'),

('Downey', 'Robert', 'IT Manager', 1, '1972-02-18 00:00:00',

'1997-06-05 00:00:00', '456 Orange Ave', 'Orlando', 'FL', 'USA', '33326',

'+1 (786) 987-6543', '+1 (786) 987-4567', 'another@gmail.com');

Task – Insert two new records into Customer table

**INSERT** **INTO**

customer(first\_name, last\_name, company, address,

city, state, country, postal\_code, phone, fax, email)

**VALUES**

('Bobby', 'Grapes', 'H&M', '123 Apple St', 'Miami', 'FL', 'USA', '33015',

'+1 (786) 453-8636', **NULL**, 'bobbygrapes@gmail.com'),

('Johnny', 'McDonald', 'Burger King', '456 Pear Road', 'Hialeah', 'FL', 'USA', '33014','+1 (786) 573-6574', **NULL**, 'johnnymcdonald@gmail.com');

## 2.4 UPDATE

Task – Update Aaron Mitchell in Customer table to Robert Walter

**UPDATE** customer

**SET** first\_name = 'Robert', last\_name = 'Walter'

**WHERE** customer\_id = 32;

Task – Update name of artist in the Artist table “Creedence Clearwater Revival” to “CCR”

**UPDATE** artist

**SET** **name** = 'CCR'

**WHERE** **name** = 'Creedence Clearwater Revival';

## 2.5 LIKE

Task – Select all invoices with a billing address like “T%”

**SELECT** \* **FROM** invoice **WHERE** billing\_address **LIKE** 'T%';

## 2.6 BETWEEN

Task – Select all invoices that have a total between 15 and 50

**SELECT** \* **FROM** invoice **WHERE** total **BETWEEN** 15 **AND** 50;

Task – Select all employees hired between 1st of June 2003 and 1st of March 2004

**SELECT** \* **FROM** employee **WHERE** hire\_date **BETWEEN** '2003-06-01' **AND** '2004-03-01';

## 2.7 DE LETE

Task – Delete a record in Customer table where the name is Robert Walter (There may be constraints that rely on this, find out how to resolve them).

**ALTER** **TABLE** invoice

**DROP** **CONSTRAINT** fk\_invoice\_customer\_id;

**DELETE** **FROM** customer **WHERE** first\_name = 'Robert' **AND** last\_name = 'Walter';

# SQL Functions

In this section you will be using the Oracle system functions, as well as your own functions, to perform various actions against the database

## 3.1 System Defined Functions

Task – Create a function that returns the current time.

Task – Create a function that returns the length of a mediatype from the mediatype table

## 3.2 System Defined Aggregate Functions

Task – Create a function that returns the average total of all invoices

Task – Create a function that returns the most expensive track

## 3.3 User Defined Scalar Functions

Task – Create a function that returns the average price of invoiceline items in the invoiceline table

## 3.4 User Defined Table Valued Functions

Task – Create a function that returns all employees who are born after 1968.

# 5.0 Transactions

In this section you will be working with transactions. Transactions are usually nested within a stored procedure.

Task – Create a transaction that given a invoiceId will delete that invoice (There may be constraints that rely on this, find out how to resolve them).

Task – Create a transaction nested within a stored procedure that inserts a new record in the Customer table

# 6.0 Triggers

In this section you will create various kinds of triggers that work when certain DML statements are executed on a table.

## 6.1 AFTER/FOR

Task - Create an after insert trigger on the employee table fired after a new record is inserted into the table.

# 7.0 JOINS

In this section you will be working with combining various tables through the use of joins. You will work with outer, inner, right, left, cross, and self joins.

## 7.1 INNER

Task – Create an inner join that joins customers and orders and specifies the name of the customer and the invoiceId.

**SELECT** customer.first\_name, customer.last\_name, invoice.invoice\_id

**FROM** invoice

**INNER** **JOIN** customer **ON** invoice.customer\_id = customer.customer\_id;

## 7.2 OUTER

Task – Create an outer join that joins the customer and invoice table, specifying the CustomerId, firstname, lastname, invoiceId, and total.

**SELECT** customer.customer\_id, invoice.invoice\_id, customer.first\_name, customer.last\_name, invoice.total

**FROM** invoice

**FULL** **OUTER** **JOIN** customer **ON** invoice.customer\_id = customer.customer\_id;

## 7.3 RIGHT

Task – Create a right join that joins album and artist specifying artist name and title.

**SELECT** artist.**name**, album.title

**FROM** album

**RIGHT** **JOIN** artist

**ON** album.artist\_id = artist.artist\_id;

## 7.4 CROSS

Task – Create a cross join that joins album and artist and sorts by artist name in ascending order.

**SELECT** \* **FROM** artist

**CROSS** **JOIN** album

**ORDER** **BY** artist.**name**;

## 7.5 SELF

Task – Perform a self-join on the employee table, joining on the reportsto column.

**SELECT** a.employee\_id **AS** "Employee ID",

a.last\_name **AS** "Employee Name",

b.employee\_id **AS** "Supervisor ID",

b.last\_name **AS** "Supervisor Name"

**FROM** employee a, employee b

**WHERE** a.reports\_to = b.employee\_id;