## SQL Hands on

Explore the basic concepts of SQL.

Explore the following:

* difference between BETWEEN and IN operators
* LIKE operators
* difference between CHAR and VARCHAR2 datatype
* different types of case manipulation functions
* difference between primary key and unique constraints
* view, foreign key, index, WITH clause
* types of joins, Sub queries
* difference between SQL DELETE and SQL TRUNCATE
* Union, minus, Interact, ALIAS commands

Solve all the below problems using both **traditional SQL queries and SQLAlchemy ORM**.

Provide the SQL query for the database directly, and also implement the same solution using SQLAlchemy's ORM approach.

1.

a) Write a SQL query for the provided table to retrieve the first five unique salespeople IDs in order based on higher purchase amounts, where each salesperson's purchase amount should not exceed 2000.

b) Write a SQL query for the provided table to retrieve the first five unique salespeople IDs in order based on lower purchase amounts, where each salesperson's purchase amount should exceed 100.

**CREATE TABLE orders (**

**ord\_no INT PRIMARY KEY,**

**purch\_amt NUMERIC,**

**ord\_date DATE,**

**customer\_id INT,**

**salesman\_id INT**

**);**

**INSERT INTO orders (ord\_no, purch\_amt, ord\_date, customer\_id, salesman\_id)**

**VALUES**

**(70001, 150.5, '2012-10-05', 3005, 5002),**

**(70009, 270.65, '2012-09-10', 3001, 5005),**

**(70002, 65.26, '2012-10-05', 3002, 5001),**

**(70004, 110.5, '2012-08-17', 3009, 5003),**

**(70007, 948.5, '2012-09-10', 3005, 5002),**

**(70005, 2400.6, '2012-07-27', 3007, 5001),**

**(70008, 5760, '2012-09-10', 3002, 5001),**

**(70010, 1983.43, '2012-10-10', 3004, 5006),**

**(70003, 2480.4, '2012-10-10', 3009, 5003),**

**(70012, 250.45, '2012-06-27', 3008, 5002),**

**(70011, 75.29, '2012-08-17', 3003, 5007),**

**(70013, 3045.6, '2012-04-25', 3002, 5001);**

2.

a) Write a SQL query for the given table to retrieve details of salespeople with commissions ranging from 0.10 to 0.12.(Begin and end values are included.) Return salesman\_id, name, city, and commission.

b) Write a SQL query for the given table to retrieve avg details of commissions ranging from 0.12 to 0.14.(Begin and end values are included.)

**CREATE TABLE salesman (**

**salesman\_id INT PRIMARY KEY,**

**name VARCHAR(255),**

**city VARCHAR(255),**

**commission FLOAT**

**);**

**INSERT INTO salesman (salesman\_id, name, city, commission)**

**VALUES**

**(5001, 'James Hoog', 'New York', 0.15),**

**(5002, 'Nail Knite', 'Paris', 0.13),**

**(5005, 'Pit Alex', 'London', 0.11),**

**(5006, 'Mc Lyon', 'Paris', 0.14),**

**(5007, 'Paul Adam', 'Rome', 0.13),**

**(5003, 'Lauson Hen', 'San Jose', 0.12);**

3.

From the following table, write a SQL query to find those employees who

worked more than or equal to two jobs in the past. Return employee id.

**CREATE TABLE employee\_history (**

**employee\_id INT,**

**start\_date DATE,**

**end\_date DATE,**

**job\_id VARCHAR(50),**

**department\_id INT**

**);**

**INSERT INTO employee\_history (employee\_id, start\_date, end\_date, job\_id, department\_id)**

**VALUES**

**(102, '2001-01-13', '2006-07-24', 'IT\_PROG', 60),**

**(101, '1997-09-21', '2001-10-27', 'AC\_ACCOUNT', 110),**

**(101, '2001-10-28', '2005-03-15', 'AC\_MGR', 110),**

**(201, '2004-02-17', '2007-12-19', 'MK\_REP', 20),**

**(114, '2006-03-24', '2007-12-31', 'ST\_CLERK', 50),**

**(122, '2007-01-01', '2007-12-31', 'ST\_CLERK', 50),**

**(200, '1995-09-17', '2001-06-17', 'AD\_ASST', 90),**

**(176, '2006-03-24', '2006-12-31', 'SA\_REP', 80),**

**(176, '2007-01-01', '2007-12-31', 'SA\_MAN', 80),**

**(200, '2002-07-01', '2006-12-31', 'AC\_ACCOUNT', 90);**

4.

a) Write a SQL statement to generate a list of salesmen who either serve one or more customers or have not joined any customer yet. The customers may have placed one or more orders with an order amount equal to or exceeding 2000, and they must have a grade. Alternatively, customers may not have placed any order with the associated supplier.(Use joins)

Return *cust\_name, cust city, grade, Salesman name, ord\_no, ord\_date, purch\_amt*

Expected Output:

cust\_name |city |grade|Salesman |ord\_no|ord\_date |purch\_amt|

--------------|----------|-----|----------|------|----------|---------|

Brad Davis |New York | 200|James Hoog| 70005|2012-07-27| 2400.60|

Nick Rimando |New York | 100|James Hoog| 70008|2012-09-10| 5760.00|

Geoff Cameron |Berlin | 100|Lauson Hen| 70003|2012-10-10| 2480.40|

Nick Rimando |New York | 100|James Hoog| 70013|2012-04-25| 3045.60|

b) Write a SQL statement to generate a report with the customer name, city, order number, order date, and purchase amount for customers on the list who must have a grade and placed one or more orders. Additionally, include orders placed by customers who are neither on the list nor have a grade.(Use joins)

Expected Output:

cust\_name city ord\_no ord\_date Order Amount

Nick Rimando New York 70002 2012-10-05 65.26

Geoff Cameron Berlin 70004 2012-08-17 110.50

Brad Davis New York 70005 2012-07-27 2400.60

Nick Rimando New York 70008 2012-09-10 5760.00

Fabian Johnson Paris 70010 2012-10-10 1983.43

Geoff Cameron Berlin 70003 2012-10-10 2480.40

Jozy Altidor Moscow 70011 2012-08-17 75.29

Nick Rimando New York 70013 2012-04-25 3045.60

Graham Zusi California 70001 2012-10-05 150.50

Graham Zusi California 70007 2012-09-10 948.50

Julian Green London 70012 2012-06-27 250.45

**A Total of 3 tables are involved in this problem. Two tables were created in the previous problems.**

**salesman from problem 2,**

**orders from problem 1**

**Other table customer, create using the below queries**

**CREATE TABLE customer (**

**customer\_id INT PRIMARY KEY,**

**cust\_name VARCHAR(255),**

**city VARCHAR(255),**

**grade INT,**

**salesman\_id INT**

**);**

**INSERT INTO customer (customer\_id, cust\_name, city, grade, salesman\_id)**

**VALUES**

**(3002, 'Nick Rimando', 'New York', 100, 5001),**

**(3007, 'Brad Davis', 'New York', 200, 5001),**

**(3005, 'Graham Zusi', 'California', 200, 5002),**

**(3008, 'Julian Green', 'London', 300, 5002),**

**(3004, 'Fabian Johnson', 'Paris', 300, 5006),**

**(3009, 'Geoff Cameron', 'Berlin', 100, 5003),**

**(3003, 'Jozy Altidor', 'Moscow', 200, 5007),**

**(3001, 'Brad Guzan', 'London', NULL, 5005);**

5.

a) ‘From the following tables write a SQL query to find those employees who work for the department where the departmental allotment amount is more than Rs. 50000. Return emp\_fname and emp\_lname.

b) From the following tables write a SQL query to find the departments with the second lowest sanction amount. Return emp\_fname and emp\_lname.

**CREATE TABLE employee (**

**emp\_idno INT PRIMARY KEY,**

**emp\_fname VARCHAR(255),**

**emp\_lname VARCHAR(255),**

**emp\_dept INT**

**);**

**INSERT INTO employee (emp\_idno, emp\_fname, emp\_lname, emp\_dept)**

**VALUES**

**(127323, 'Michale', 'Robbin', 57),**

**(526689, 'Carlos', 'Snares', 63),**

**(843795, 'Enric', 'Dosio', 57),**

**(328717, 'Jhon', 'Snares', 63),**

**(444527, 'Joseph', 'Dosni', 47),**

**(659831, 'Zanifer', 'Emily', 47),**

**(847674, 'Kuleswar', 'Sitaraman', 57),**

**(748681, 'Henrey', 'Gabriel', 47),**

**(555935, 'Alex', 'Manuel', 57),**

**(539569, 'George', 'Mardy', 27),**

**(733843, 'Mario', 'Saule', 63),**

**(631548, 'Alan', 'Snappy', 27),**

**(839139, 'Maria', 'Foster', 57);**

**CREATE TABLE department (**

**dpt\_code INT PRIMARY KEY,**

**dpt\_name VARCHAR(255),**

**dpt\_allotment INT**

**);**

**INSERT INTO department (dpt\_code, dpt\_name, dpt\_allotment)**

**VALUES**

**(57, 'IT', 65000),**

**(63, 'Finance', 15000),**

**(47, 'HR', 240000),**

**(27, 'RD', 55000),**

**(89, 'QC', 75000);**