SQL COURSE CODES (With Headings Based on Video Chapters)

https://www.youtube.com/watch?v=7S_tz1z_5bA

📌 The SELECT Statement (0:23:40)

```
SELECT * FROM customers;
```

📌 The SELECT Clause (0:29:30)

```
SELECT
   first_name,
   last_name,
   points,
   (points + 10) * 100 AS 'discount factor'
```

```
FROM customers;

SELECT state FROM customers;
SELECT DISTINCT state FROM customers;

SELECT name, unit_price, (unit_price * 1.1) AS 'new price'
FROM Products;
```

📌 The WHERE Clause (0:38:18)

```
SELECT * FROM customers
WHERE birth_date > '1990-01-01';
```

📌 The AND, OR, and NOT Operators (0:43:35)

```
SELECT * FROM Customers
WHERE birth_date > '1990-01-01' OR
    (points > 1000 AND state = 'VA');

SELECT * FROM Customers
WHERE NOT (birth_date > '1990-01-01' OR points > 1000);
```

📌 The IN Operator (0:51:38)

```
SELECT * FROM customers
WHERE state IN ('VA', 'GA', 'FL');

SELECT * FROM products
WHERE quantity_in_stock IN (49, 38, 72);
```

📌 The BETWEEN Operator (0:54:41)

```
SELECT * FROM customers
WHERE points BETWEEN 1000 AND 3000;

SELECT * FROM customers
WHERE birth_date BETWEEN '1990-01-01' AND '2000-01-01';
```

📌 The LIKE Operator (0:56:53)

```
SELECT * FROM customers
WHERE last_name LIKE 'b____y';

SELECT * FROM customers
WHERE phone LIKE '%9%' AND (address LIKE '%TRAIL' OR address LIKE '%AVENUE');

SELECT * FROM customers
WHERE last_name LIKE '%field%';
```

📌 The REGEXP Operator (1:02:31)

```
SELECT * FROM customers
WHERE last_name REGEXP 'field$|mac|rose';

SELECT * FROM customers
WHERE last_name REGEXP '[a-h]e';

SELECT * FROM customers
WHERE first_name REGEXP 'ELKA|AMBUR';

SELECT * FROM customers
WHERE last_name REGEXP 'EY$|ON$';

SELECT * FROM customers
```

```
WHERE last_name REGEXP '^MY|^SE';

SELECT * FROM customers
WHERE last_name REGEXP 'b[RU]';
```

📌 The IS NULL Operator (1:11:51)

```
SELECT * FROM customers
WHERE phone IS NULL;

SELECT * FROM orders
WHERE shipped_date IS NULL;
```

📌 The ORDER BY Operator (1:14:18)

```
SELECT * FROM customers
ORDER BY state DESC, first_name DESC;

SELECT first_name, last_name, 10 AS points
FROM customers
ORDER BY 1 ,2 ;

SELECT *, quantity * unit_price AS total_price
FROM order_items
WHERE order_id = 2
ORDER BY total_price DESC;
```

📌 The LIMIT Operator (1:21:23)

```
SELECT * FROM customers
LIMIT 6 , 3;
```

```
SELECT * FROM customers
ORDER BY points DESC
LIMIT 3;
```

📌 Inner Joins (1:24:50)

```
SELECT order_id,  o.customer_id, first_name, last_name
FROM orders o
JOIN customers c ON o.customer_id = c.customer_id;

SELECT order_id, oi.product_id, quantity, oi.unit_price
FROM order_items oi
JOIN products p ON p.product_id = oi.product_id;
```

📌 Joining Across Databases (1:33:16)

```
USE sql_inventory;
SELECT *
FROM order_items oi
JOIN sql_inventory.products p ON oi.product_id = p.product_id;
```

📌 Self Joins (1:36:03)

```
USE sql_hr;
SELECT e.employee_id, e.first_name, e.last_name, h.first_name, h.last_name
FROM sql_hr.employees e
JOIN sql_hr.employees h ON e.reports_to = h.employee_id;
```

📌 Joining Multiple Tables (1:40:17)

```
USE sql_invoicing;
SELECT i.invoice_id, i.number, i.client_id, c.name, i.invoice_date, i.due_date, p.amount,
p.date AS Paid
FROM invoices i
JOIN clients c ON i.client_id = c.client_id
JOIN payments p ON i.payment_date = p.date;
```

📌 The USING Clause (2:05:50)

```
SELECT i.invoice_id, i.number, i.client_id, c.name AS Client, i.invoice_total, i.invoice_date,
i.payment_date
FROM invoices i
JOIN clients c USING (client_id)
WHERE payment_date IS NOT NULL;
```

📌 Inserting a Single Row (2:29:54)

```
USE sql_store;
INSERT INTO customers (first_name, last_name, birth_date, address, city, state)
VALUES ('John', 'Smith', '1990-01-01', 'address', 'city', 'CA');
```

📌 Inserting Multiple Rows (2:35:40)

```
INSERT INTO shippers (name)
VALUES ('shipper1'), ('shipper2'), ('shipper3');

INSERT INTO products
VALUES (22, 'harsh', 80, 2.2), (23, 'harsha', 80, 2.2), (24, 'harshaa', 80, 2.2);
```

📌 Inserting Hierarchical Rows (2:38:58)

```
INSERT INTO orders (customer_id, order_date, status)
VALUES (1, '2019-01-02', 1);

INSERT INTO order_items
VALUES (LAST_INSERT_ID(), 1, 1, 2.95), (LAST_INSERT_ID(), 2, 1, 3.95);
```

📌 Creating a Copy of a Table (2:44:51)

```
CREATE TABLE orders_archived AS
SELECT * FROM orders;

INSERT INTO orders_archived
SELECT * FROM orders
WHERE order_date < '2019-01-01';

CREATE TABLE Invoices_Archived AS
SELECT i.invoice_id, i.number, i.client_id, c.name, i.invoice_date, i.due_date, p.amount,
p.date AS Paid
FROM invoices i
JOIN clients c ON i.client_id = c.client_id
JOIN payments p ON i.payment_date = p.date;
```

📌 Updating a Single Row (2:53:38)

```
UPDATE invoices
SET payment_total = invoice_total * 0.5, payment_date = due_date
WHERE invoice_id = 3;
```

📌 Updating Multiple Rows (2:57:33)

```
UPDATE invoices
SET payment_total = invoice_total * 0.5, payment_date = due_date
WHERE client_id IN (3, 4);
```

📌 Using Subqueries in Updates (3:00:47)

```
UPDATE invoices
SET payment_total = invoice_total * 0.5, payment_date = due_date
WHERE client_id = (
  SELECT client_id FROM clients WHERE state IN ('CA', 'NY')
);

UPDATE orders
SET comments = 'GOLD CUSTOMERS'
WHERE customer_id IN (
  SELECT customer_id FROM customers WHERE points > 300
);
```

📌 Deleting Rows (3:06:24)

```
DELETE FROM orders
WHERE order_date < '2018-01-01';

DELETE FROM customers
WHERE phone IS NULL;
```

📌 Restoring Course Databases (3:07:48)

```
-- Restore from backup SQL script
SOURCE 'C:/path_to_backup/backup.sql';

-- OR using MySQL CLI:
```

```
-- mysql -u username -p sql_store < backup.sql
```