ALTER TABLE transactions --**continuation of reviewer 4**

ADD CONSTRAINT fk\_customer\_id

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)

**ON DELETE SET NULL**;

>

DELETE FROM customers

WHERE customer\_id = 4; >deleting a value and setting to null<

ALTER TABLE transactions

ADD CONSTRAINT fk\_transactions\_id

FOREIGN KEY(customer\_id) REFERENCES customers(customer\_id)

**ON DELETE CASCADE**; >deleting an entire row<

>

DELETE FROM customers

WHERE customer\_id = 4;

-- **Stored procecure** = is prepared SQL code that you can save

great if there's a query that you write often

>

**DELIMITER** $$ >setting the stopper(;) to a another symbol

**CREATE PROCEDURE** get\_customers() >creating a procedure

**BEGIN**

SELECT \* FROM customers; >tab is necessary

**END** $$

DELIMITER ;

>

**CALL** get\_customers(); >execute the code within the stored procedure

**DROP PROCEDURE** get\_customers; >droppping procedure

DELIMITER $$

CREATE PROCEDURE find\_customer(IN id INT)

BEGIN

SELECT \*

FROM customers

WHERE customer\_id = id;

END $$

DELIMITER ;

>

**CALL find\_customer(1)**; >finding the customer tru id

DELIMITER $$

CREATE PROCEDURE find\_customer(IN f\_name VARCHAR(50),

IN l\_name VARCHAR(50))

BEGIN

SELECT \*

FROM customers

WHERE first\_name = f\_name AND last\_name = l\_name;

END $$

DELIMITER ;

>

CALL find\_customer("Larry", "Lobster");

>finding customer true first and last name

**Trigger** = when an event happens, ao something

ex. (**INSERT, UPDATE, DELETE**)

checks data, handles errors, auditing tables

**CREATE TRIGGER** before\_hourly\_pay\_update >creating a trigger

**BEFORE UPDATE ON** employees >before/after -update,insert,delete

**FOR EACH ROW** >one or more rows

SET salary = (hourly\_pay \* 2080);

#**SET NEW.salary** if new **NEW.hourly\_pay** (same)

> >automatically setting the salary when the h.pay update

**SHOW TRIGGERS;** >show the triggers and their info

CREATE TRIGGER before\_hourly\_pay\_insert

**BEFORE INSERT ON** employees

FOR EACH ROW

SET NEW.salary = (NEW.hourly\_pay \* 2080);

>

INSERT INTO employees

VALUES(6, "Sheldon", "Plankton", 10, NULL, "janitor", "2023-01-07", 5);

SELECT \* FROM employees;

CREATE TRIGGER after\_salary\_delete

**AFTER DELETE ON** employees

FOR EACH ROW

UPDATE expenses >decreasing salaries/values

SET expense\_total = expense\_total **- OLD.**salary

WHERE expense name = "salaries";

>

DELETE FROM employees

WHERE employee\_id = 6;

CREATE TRIGGER after\_salary\_insert

**AFTER INSERT ON** employees

FOR EACH ROW

UPDATE expenses >increasing salaries

SET expense\_total = expense\_total + NEW.salary

WHERE expense name = "salaries";

CREATE TRIGGER

AFTER UPDATE ON employees

FOR EACH ROW

UPDATE expenses

SET expense\_total = expense\_total + (NEW.salary - OLD.salary)

=WHERE expense\_name = "salaries";

>

UPDATE employees

SET hourly\_pay = 10

WHERE employee\_id = 1

SELECT \* FROM expenses; >difference of the salaries