1. ANY, GROUP BY, Aggregations

SELECT prod\_name

FROM products

WHERE prod\_id = ANY

(SELECT prod\_id, COUNT(\*)

FROM order\_details

HAVING COUNT(\*) > 100

GROUP BY prod\_id)

Вывести название товара который были заказан более 100 раз

1. IN

SELECT \*

FROM departments

WHERE department\_id IN

(SELECT department\_id

FROM employees

WHERE YEAR(employee\_date) <= (YEAR(GETDATE()) -1)

Вывести номер и описание отдела сотрудников которые начали работать больше года назад

1. EXCEPT

SELECT customer\_id

FROM customers

EXCEPT

SELECT customer\_id

FROM orders

Вывести клиентов которые ни разу не делали заказ

1. ALL РОБИТ

SELECT prod\_name

FROM products

WHERE prod\_id = ALL

(SELECT prod\_id

FROM order\_details

WHERE order\_quantity > 5);

Вывести названия продуктов которые были заказаны более 5 раз в одном заказе

1. UPDATE TRIGGER

CREATE TRIGGER discount

AFTER UPDATE ON products\_info

WHEN prod\_num < 10

UPDATE products\_info

SET price = price – 1

1. UNION

(SELECT a.supp\_id

FROM suppliers AS a

WHERE supp\_name LIKE ‘A%’)

UNION

(SELECT b.supp\_id

FROM products\_info AS b

GROUP BY b.supp\_id

HAVING COUNT(\*) > 5)

1. INTERSECT

(SELECT cust\_id, order\_id, location

FROM customers AS a

LEFT JOIN orders AS b

ON a.cust\_id = b.cust\_id)

INTERSECT

(SELECT cust\_id, order\_id, location

FROM customers AS a

RIGHT JOIN orders AS b

ON a.cust\_id = b.cust\_id)

1. DELETE

DELETE products

FROM products AS a

JOIN best\_products AS b on a.prod\_id = b.prod\_id

WHERE b.sold = 0

1. UPDATE

UPDATE products\_info

SET price = price + 2

WHERE supp\_id = ALL

(SELECT supp\_id

FROM suppliers

WHERE supp\_name LIKE ‘Pepsi%’)

1. EXISTS

SELECT cust\_id

FROM customers AS a

WHERE EXISTS

(SELECT order\_id

FROM orders AS b

WHERE a.cust\_id = b.cust\_id)

1. INSERT TRIGGER

CREATE TRIGGER order

AFTER INSERT ON order\_decline

WHERE orders.order\_id = order\_decline.order\_id

DELETE FROM orders

1. NULL

SELECT a.shipping\_id, a.cust\_id, b.name

FROM shipping\_details AS a, customers AS b

WHERE a.cust\_id = b.cust\_id

AND a.address IS NULL

1. ALTER

ALTER TABLE reviews

ADD rating int

1. CREATE VIEW review AS

SELECT prod\_name, review

FROM products, reviews

WHERE products.prod\_id = reviews.prod\_id;

1. DELETE TRIGGER

CREATE TRIGGER update\_order

AFTER DELETE ON employees

WHERE orders.employee\_id = employee.employee\_id

UPDATE orders

SET employee\_id = NULL;

1. RENAME

ALTER TABLE best\_products

RENAME COLUMN sold TO totalSales;

1. Function

SELECT MAX(price) AS max\_price

FROM products;

18 ALTER TABLE Employees

ADD CONSTRAINT d\_Employee

DEFAULT 'Employee' FOR EMPLOYEE\_NAME;

19 select products.prod\_name, best\_products.sold from products, best\_products

where products.prod\_id = best\_products.prod\_id

order by best\_products.sold desc

20 SELECT supp\_id FROM

Products JOIN best\_products ON

Products.prod\_id = best\_products.prod\_id

WHERE sold > 80

TRANSACTION AND INDEX

BEGIN TRANSACTION

SELECT FROM products

WHERE prod\_id = products.prod\_id

IF query result is empty then

INSERT INTO products VALUES(prod\_id, cat\_id, prod\_name)

COMMIT

ELSE

ROLLBACK

CREATE INDEX product\_name\_idx

ON products(prod\_name)