**DML- команды в**

1. **INSERT – вставляет в таблицу строку**

INSERT INTO countries VALUES

('SW','Sweden', 1) – добавляет значения по порядку столбцов

INSERT INTO countries *(country\_id, country\_name, region\_id)* values

('NR','Norway',1) – добавляет значения в соответствии с порядком столбцов

insert into employees (employee\_id, last\_name, email, hire\_date, job\_id) values

(211, INITCAP('tregulov'), UPPER('TREGULOV1'),to\_date('18-SEP-2019','dd-MON-YYYY'), UPPER('IT\_PROG')) – использование функции чтоб все данные были унифицированными.

insert into new\_emps (emp\_id, name, start\_date)

select employee\_id, first\_name, hire\_date from employees where employee\_id > 200 – использование подзапроса для вставки данных в другую таблицу.

insert into new\_emps

select employee\_id, first\_name, hire\_date, job\_id from employees where employee\_id > 200 – без указания наименования столбцов. Нужно чтоб в подзапросе было столько же столбцов сколько в таблице, куда будет вставляться.

**INSERT ALL** – вставляет данные в несколько таблиц

when department\_id = 100 then

INTO emps\_with\_dept\_100 values (first\_name, salary) – вставляет имя и зарплату людей у которых dep\_id = 100

when salary>15000 then

INTO emps\_with\_high\_salary (name) values (last\_name) – вставляет фамилию людей с зарплатой больше 15000

when 5=5 then

into some\_emps (name, salary) values (first\_name, salary)

select first\_name, last\_name, salary, department\_id from employees

where length(first\_name) > 5

1. **UPDATE – изменяет информацию в уже существующих строках или столбцах**

*UPDATE* employees *SET* salary =25000

where employee\_id=100;

update employees set salary = 5000

where department\_id in (*select department\_id from departments where department\_name='Marketing'*) – использование подзапроса

update employees set salary = (select max(salary) from employees), hire\_date = (select min(start\_date) from job\_history)

where employee\_id = 180;

1. **DELETE – удаляет строки из таблицы**

DELETE from employees

where employee\_id = 210

delete from new\_emps

where job in (select distinct job\_id from employees where department\_id

in (select department\_id from departments where manager\_id = 100));

1. **MERGE – совмещает функциональность INSERT, UPDATE, DELETE**

MERGE INTO new\_emps ne

USING employees e

ON (ne.emp\_id = e.employee\_id)

WHEN MATCHED THEN

UPDATE SET ne.start\_date = SYSDATE

DELETE where ne.job like '%IT%'

WHEN NOT MATCHED THEN

INSERT (emp\_id, name, start\_date, job) values (employee\_id, last\_name, hire\_date, job\_id);

**ТРАНЗАКЦИИ. ПРИНЦИПЫ ACID**

**TCL – transaction control language.**

**COMMIT – команда которая подтверждает изменения внесенные в таблицу**

**ROLLBACK – можно отменить все изменения в одной транзакции до последнего commit.**

ROLLBACK to SAVEPOINT savepoint\_name

**SAVEPOINT – необходим для ROLLBACK. При rollback откатывает изменения до savepoint.**

SAVEPOINT savepoint\_name

select \* from new\_emps *for update* - for update позволяет залочить выбранные строки, чтоб другой пользователь не мог их изменить.