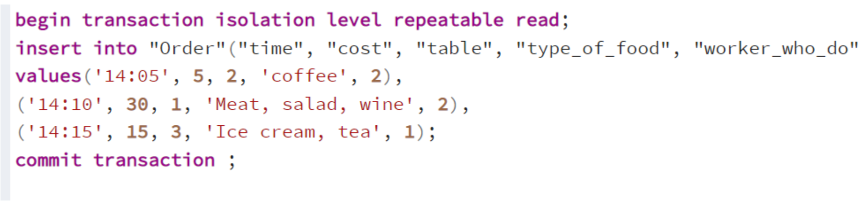
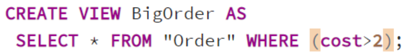
## Transaction - conflict if T1 reads orders while T2 writes them

```
COMMIT
```

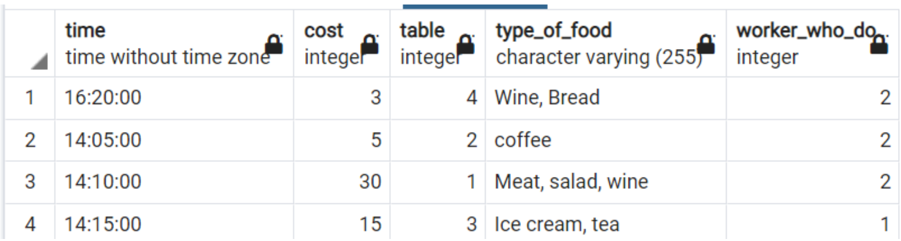


```
begin transaction isolation level repeatable read;
insert into "Order"("time", "cost", "table", "type_of_food", "worker_who_do"
values('14:05', 5, 2, 'coffee', 2),
('14:10', 30, 1, 'Meat, salad, wine', 2),
('14:15', 15, 3, 'Ice cream, tea', 1);
commit transaction ;
```

## View – big order

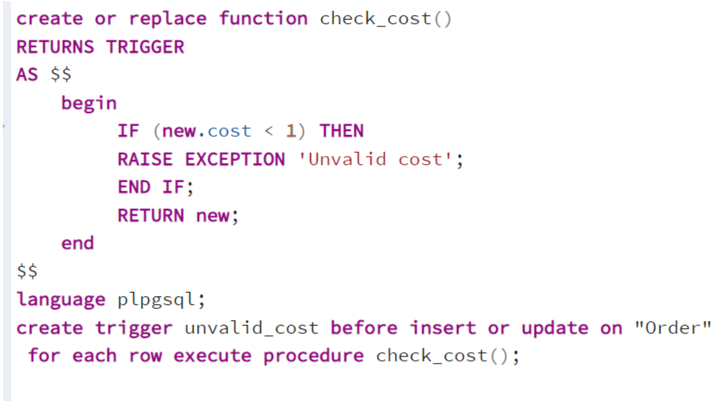


```
CREATE VIEW BigOrder AS
 SELECT * FROM "Order" WHERE (cost>2);
```

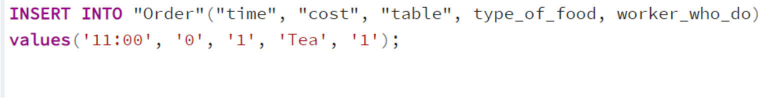


|  | time<br>time without time zone | cost<br>integer | table<br>integer | type_of_food<br>character varying (255) | worker_who_do<br>integer |
|---|---|---|---|---|---|
| 1 | 16:20:00 | 3 | 4 | Wine, Bread | 2 |
| 2 | 14:05:00 | 5 | 2 | coffee | 2 |
| 3 | 14:10:00 | 30 | 1 | Meat, salad, wine | 2 |
| 4 | 14:15:00 | 15 | 3 | Ice cream, tea | 1 |

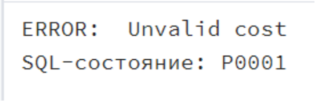
## Trigger – if cost smaller than 1



```
create or replace function check_cost()
RETURNS TRIGGER
AS $$
    begin
         IF (new.cost < 1) THEN
         RAISE EXCEPTION 'Unvalid cost';
         END IF;
         RETURN new;
    end
$$
language plpgsql;
create trigger unvalid_cost before insert or update on "Order"
 for each row execute procedure check_cost();
```

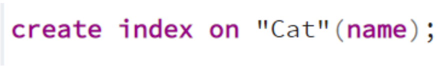


```
INSERT INTO "Order"("time", "cost", "table", type_of_food, worker_who_do)
values('11:00', '0', '1', 'Tea', '1');
```

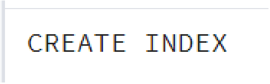


```
ERROR:  Unvalid cost
SQL-состояние: P0001
```

## Index on Cat name



```
create index on "Cat"(name);
```



```
CREATE INDEX
```