|  |
| --- |
|  |
|  | Task 1  Large objects (photos, videos, CAD files, etc.) are stored as  a large object: |
|  | • blob: binary large object -- object is a large collection of |
|  | uninterpreted binary data (whose interpretation is left to |
|  | an application outside of the database system) |
|  | • clob: character large object -- object is a large collection |
|  | of character data |
|  | When a query returns a large object, a pointer is returned |
|  | rather than the large object itself. |

|  |
| --- |
|  |
|  |
|  | Task 2  Privileges control the ability to run SQL statements.  A role is a group of privileges. |
|  | Granting a role to a user gives them the privileges contained in the role. |
|  |  |
|  | Granting a privilege on a view does not imply granting any |
|  | privileges on the underlying relations. |
|  | ♣ The grantor of the privilege must already hold the privilege |
|  | on the specified item (or be the database administrator). |
|  |  |
|  | A role is a way to distinguish among various users as far |
|  | as what these users can access/update in the database. |
|  |  |
|  | Once a role is created we can assign “users” to the role. |

--2  
CREATE ROLE accountant;  
CREATE ROLE administrator;  
CREATE ROLE support;  
-- Drop ROLE accountant;  
-- Drop ROLE administrator;  
-- Drop ROLE support;  
GRANT SELECT ON transactions TO accountant;  
GRANT SELECT, UPDATE (balance) ON accounts TO accountant;  
GRANT ALL PRIVILEGES ON SCHEMA lab7 TO administrator;  
GRANT SELECT ON accounts,customers,transactions TO support;  
select \* FROM pg\_roles;  
CREATE USER Eren;  
CREATE USER Armin;  
CREATE USER Mikasa;  
-- DROP USER Eren;  
-- DROP USER Armin;  
-- DROP USER Mikasa;  
  
GRANT accountant TO Armin;  
GRANT administrator TO Eren;  
GRANT support TO Mikasa;  
  
CREATE ROLE role\_manager CREATEROLE;  
GRANT role\_manager to Eren;  
  
REVOKE UPDATE ON accounts FROM accountant;  
--3  
SELECT \* FROM transactions;  
ALTER TABLE transactions  
-- ALTER COlUMN \* SET NOT NULL;  
ALTER COLUMN date SET NOT NULL;  
ALTER TABLE transactions  
ALTER COLUMN src\_account SET NOT NULL;  
ALTER TABLE transactions  
ALTER COLUMN dst\_account SET NOT NULL;  
ALTER TABLE transactions  
ALTER COLUMN amount SET NOT NULL;  
ALTER TABLE transactions  
ALTER COLUMN status SET NOT NULL;  
  
--4  
CREATE TYPE Valuta as (sql char(3));  
ALTER TABLE accounts  
ALTER COLUMN currency TYPE Valuta USING currency :: valuta;  
-- drop type valuta;  
--5  
CREATE UNIQUE INDEX idx\_acc ON accounts(customer\_id, currency);  
CREATE INDEX search\_t ON accounts(currency, balance);  
--6  
DO  
$$  
 DECLARE  
 bal INT;  
 lim INT;  
 BEGIN  
 UPDATE transactions SET status = 'init'  
 WHERE id = 2;  
 UPDATE accounts  
 SET balance = balance - 400  
 WHERE account\_id = 'NK90123';  
 UPDATE accounts  
 SET balance = balance + 400  
 WHERE account\_id = 'AB10203';  
 SELECT balance INTO bal FROM accounts  
 WHERE account\_id = 'NK90123';  
 SELECT accounts.limit INTO lim FROM accounts  
 WHERE account\_id = 'NK90123';  
 IF bal < lim THEN  
 UPDATE transactions SET status = 'rollback' WHERE id = 2;  
 ELSE  
 COMMIT;  
 UPDATE transactions SET status = 'commited' WHERE id = 2;  
 END IF;  
 END  
$$