# B0B36DBS: **Database Systems** | Class 8: **SQL: Advanced Constructs**

---

## Schema

```
CREATE TABLE accounts (
   ida INT PRIMARY KEY,
   number VARCHAR(22) NOT NULL UNIQUE,
   owner VARCHAR(100) NOT NULL,
   city VARCHAR(50) NOT NULL,
   balance DECIMAL(15, 2) DEFAULT 0 NOT NULL
);


CREATE TABLE transfers (
   idt BIGINT PRIMARY KEY,
   datetime TIMESTAMP NOT NULL,
   source INT REFERENCES accounts (ida) ON DELETE SET NULL,
   target INT REFERENCES accounts (ida) ON DELETE SET NULL,
   amount DECIMAL(15, 2) NOT NULL
);
```

---

## 01: Insert

```
INSERT INTO accounts
VALUES
   (501, '123456789/1111', 'Martin Svoboda', 'Liberec', DEFAULT),
   (502, '101010101/1111', 'Irena Mlynkova, 'Praha', DEFAULT);


INSERT INTO accounts (ida, number, owner, city)
VALUES
   (501, '123456789/1111', 'Martin Svoboda', 'Liberec'),
   (502, '101010101/1111', 'Irena Mlynkova, 'Praha');
```

---

## 02: Update

```
UPDATE accounts
SET owner = 'Irena Holubova', city = 'Praha'
WHERE (ida = 502);


UPDATE accounts
SET balance = balance * 1.01
WHERE (city = 'Liberec');
```

---

## 03: Delete

```
DELETE FROM accounts
WHERE (number = '101010101/1111');


DELETE FROM accounts;
```

## Data

```
INSERT INTO accounts
VALUES
   (501, '123456789/1111', 'Martin Svoboda', 'Liberec', 15000.00),
   (502, '101010101/1111', 'Irena Holubova', 'Praha', 20000.00),
   (503, '111222333/1111', 'Jiri Helmich', 'Liberec', 5000.00),
   (504, '444555666/1111', 'Martin Necasky', 'Jicin', 15000.00),
   (505, '777888999/1111', 'Marek Polak', 'Praha', 5000.00);


INSERT INTO transfers
VALUES
   (10000034, '2017-01-15 14:30:00', 501, 502, 5000.00),
   (10000035, '2017-01-15 14:40:00', 502, 503, 1000.00),
   (10000036, '2017-01-15 14:50:00', 503, 504, 2000.00),
   (10000037, '2017-01-15 15:00:00', 503, 505, 3000.00),
   (10000038, '2017-01-15 15:10:00', 501, 502, 1000.00),
   (10000039, '2017-01-15 15:20:00', 501, 504, 5000.00);
```

## 04+05: View

```
CREATE VIEW reichenberg AS
   SELECT *
   FROM accounts
   WHERE (balance >= 10000) AND (city = 'Liberec');


… WITH LOCAL CHECK OPTION;


… WITH CASCADED CHECK OPTION;


… WITH CHECK OPTION;


INSERT INTO reichenberg
VALUES (506, '999888777/1111', 'Jakub Klimek', 'Liberec', 5000.00);


INSERT INTO reichenberg
VALUES (507, '666555444/1111', 'Jakub Lokoc', 'Brno', 15000.00);
```

## 06: Explain

```
EXPLAIN
   SELECT *, (SELECT COUNT(*) FROM transfers WHERE source = ida) AS transfers
   FROM accounts
   WHERE (balance < (SELECT AVG(balance) FROM accounts)) AND (city = 'Liberec');
```

## 07: Index

```
CREATE INDEX accounts_ix_city ON accounts (city);


CREATE INDEX accounts_ix_balance ON accounts (balance);


CREATE INDEX accounts_ix_balance_city ON accounts (balance, city);
```

## 08: Procedure

```
CREATE FUNCTION transfer(i BIGINT, s INT, t INT, a DECIMAL(15, 2))
RETURNS BOOLEAN
AS $$
  DECLARE
    b1 DECIMAL(15, 2);
    b2 DECIMAL(15, 2);
  BEGIN
    b1 := (SELECT balance FROM accounts WHERE (ida = s));
    b2 := (SELECT balance FROM accounts WHERE (ida = t));
    IF ((b1 IS NULL) OR (b2 IS NULL) OR (b1 < a)) THEN RETURN false; END IF;
    UPDATE accounts SET balance = balance - a WHERE (ida = s);
    UPDATE accounts SET balance = balance + a WHERE (ida = t);
    INSERT INTO transfers VALUES (i, NOW(), s, t, a);
    RETURN true;
  END;
$$
LANGUAGE plpgsql;


SELECT transfer(10000040, 501, 502, 1000.00);
```

## 09: Transaction

```
BEGIN TRANSACTION ISOLATION LEVEL SERIALIZABLE;
SELECT transfer(10000040, 501, 502, 1000.00);
COMMIT TRANSACTION;
```

## 10: Trigger

```
CREATE FUNCTION check_balance()
RETURNS TRIGGER
AS $$
  IF ((NEW.balance IS NULL) OR (NEW.balance < 0)) THEN
    RAISE EXCEPTION 'Invalid balance value';
  END IF;
  RETURN NEW;
$$
LANGUAGE plpgsql;


CREATE TRIGGER account_tg_balance BEFORE INSERT OR UPDATE ON accounts
FOR EACH ROW EXECUTE PROCEDURE check_balance();
```