DataBaeses

UQU PAY

Supervised by:
Dr.Hassan Al-Thabti

Taple Of Content

Introduction

EERD & Entites

Database Schema

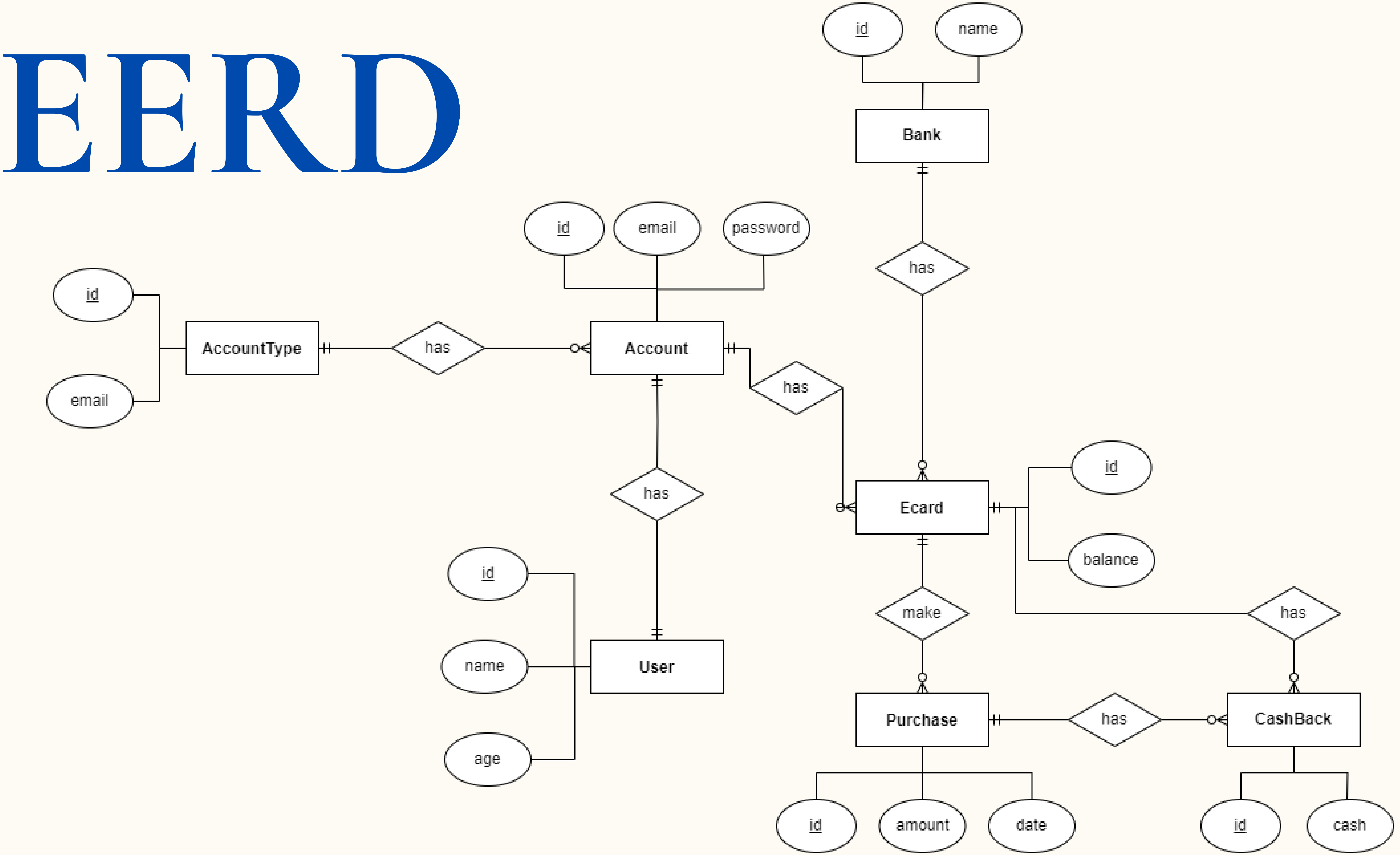
Queries

Normalization

Conclusion

Introduction

EERD


id
name
Bank
has
id
email
password
id
AccountType
email
has
Account
has
has
Ecard
id
balance
id
name
User
age
make
has
Purchase
has
CashBack
id
amount
date
id
cash

Entites

User (id, name, age).

Account (id, email, password, type, user_id, account_id).

AccountType: (id, email).

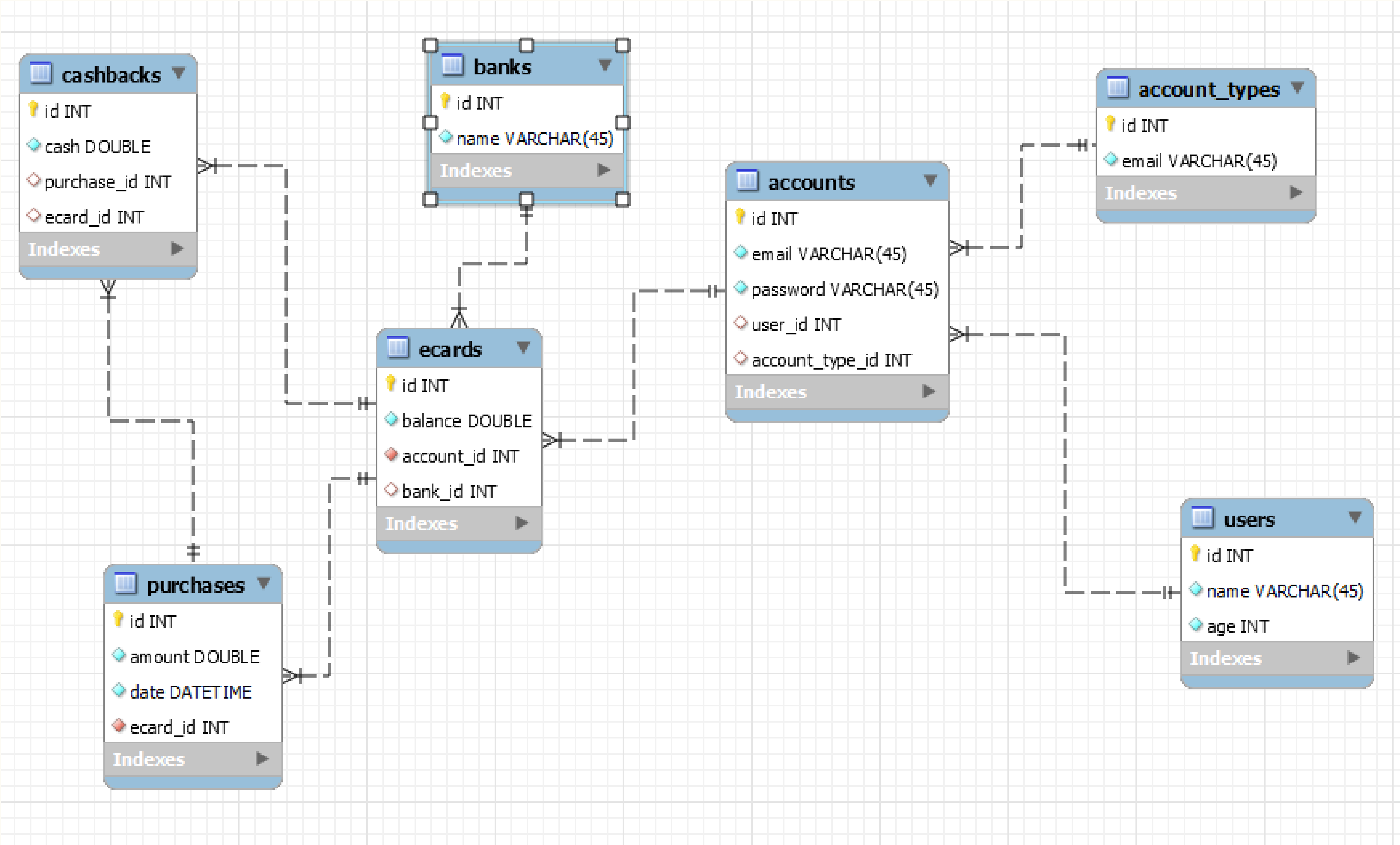
BANK (id, name).

Ecard (id, balance, account_id, bank_id).

Purchase (id, amount, date, ecard_id).

CashBack (id, cash, purchase_id, ecard_id)

Database Schema



cashbacks
id INT
cash DOUBLE
purchase_id INT
ecard_id INT
Indexes

banks
id INT
name VARCHAR(45)
Indexes

account_types
id INT
email VARCHAR(45)
Indexes

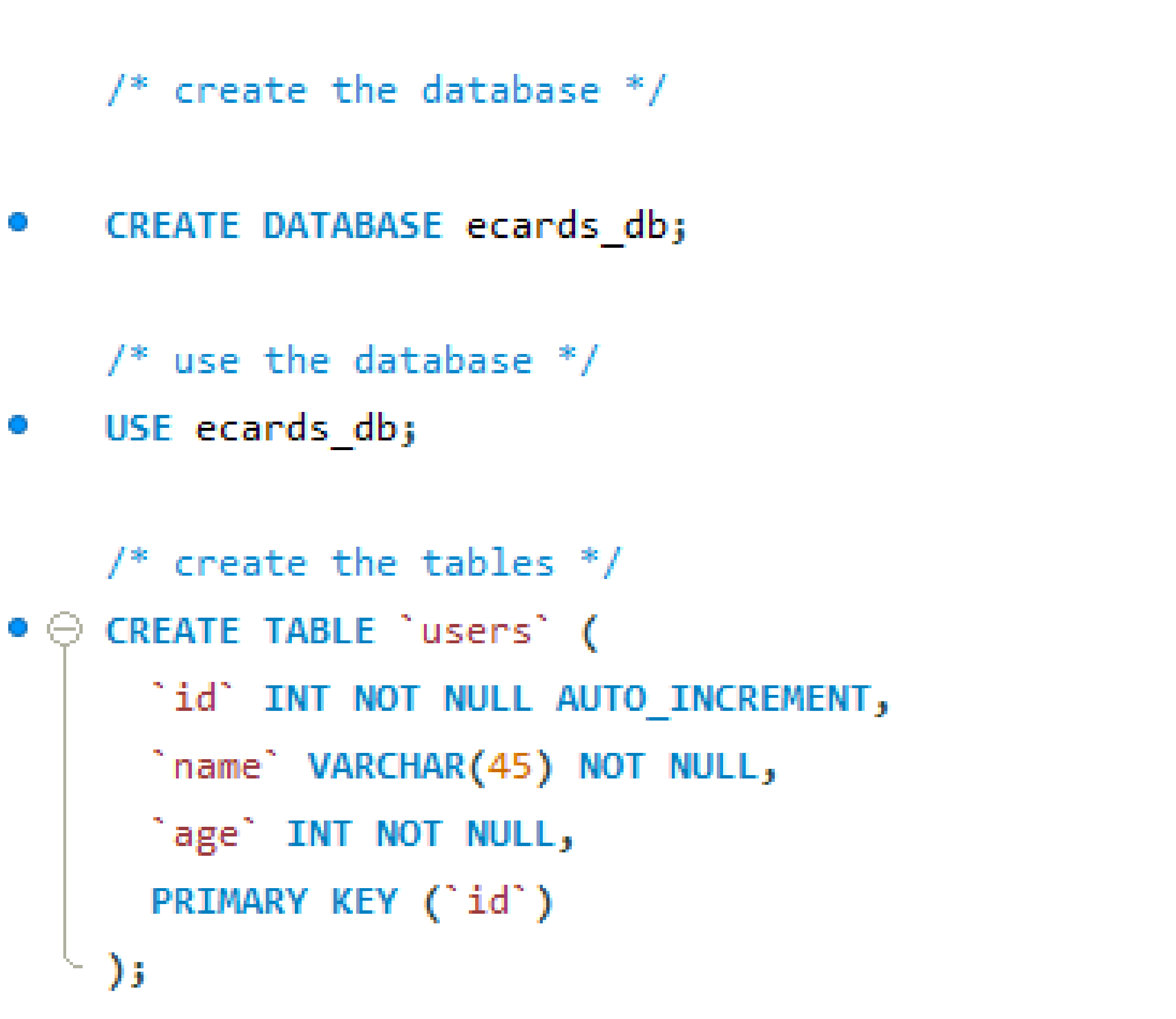
accounts
id INT
email VARCHAR(45)
password VARCHAR(45)
user_id INT
account_type_id INT
Indexes

ecards
id INT
balance DOUBLE
account_id INT
bank_id INT
Indexes

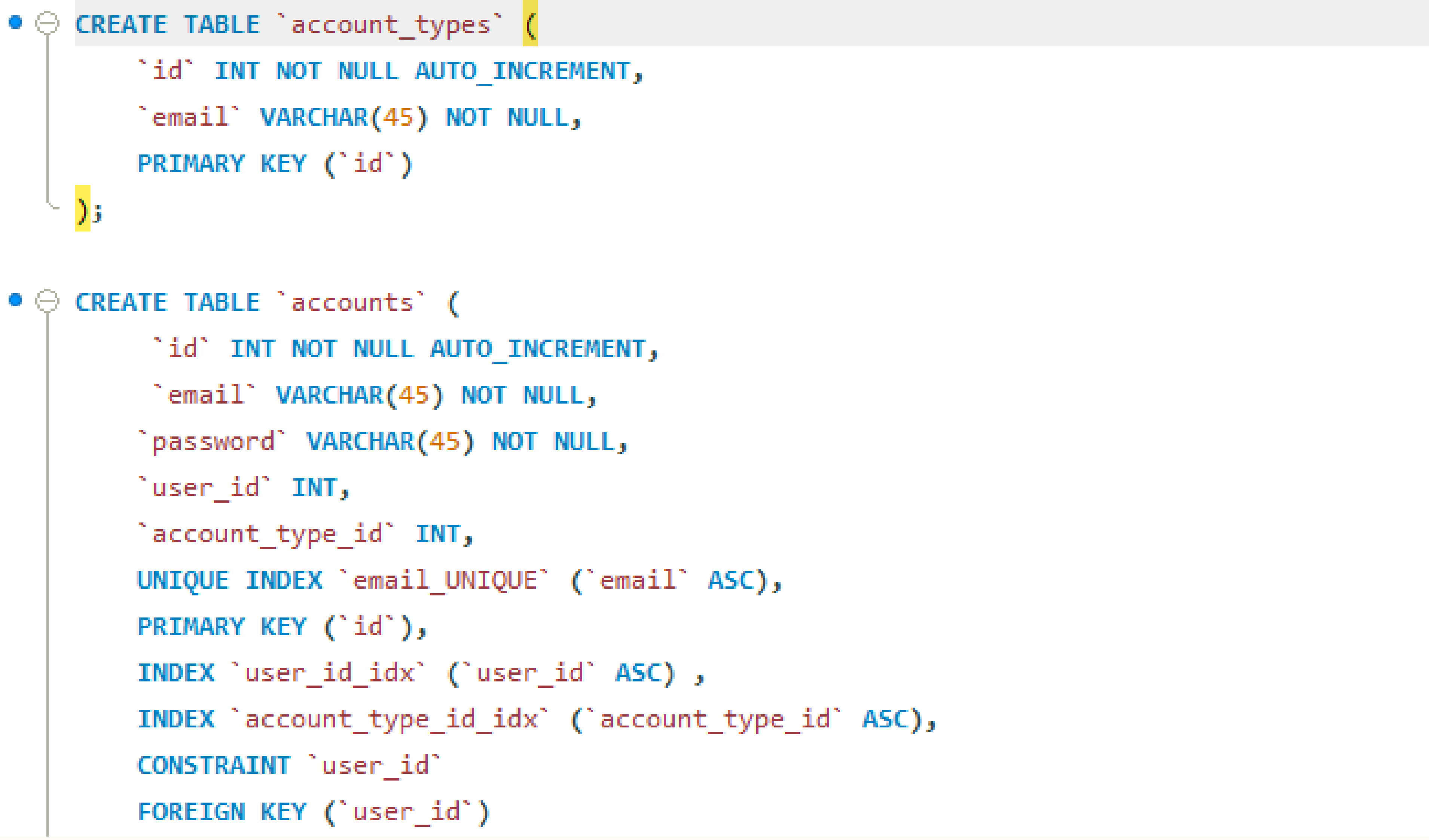
users
id INT
name VARCHAR(45)
age INT
Indexes

purchases
id INT
amount DOUBLE
date DATETIME
ecard_id INT
Indexes

Database Schema

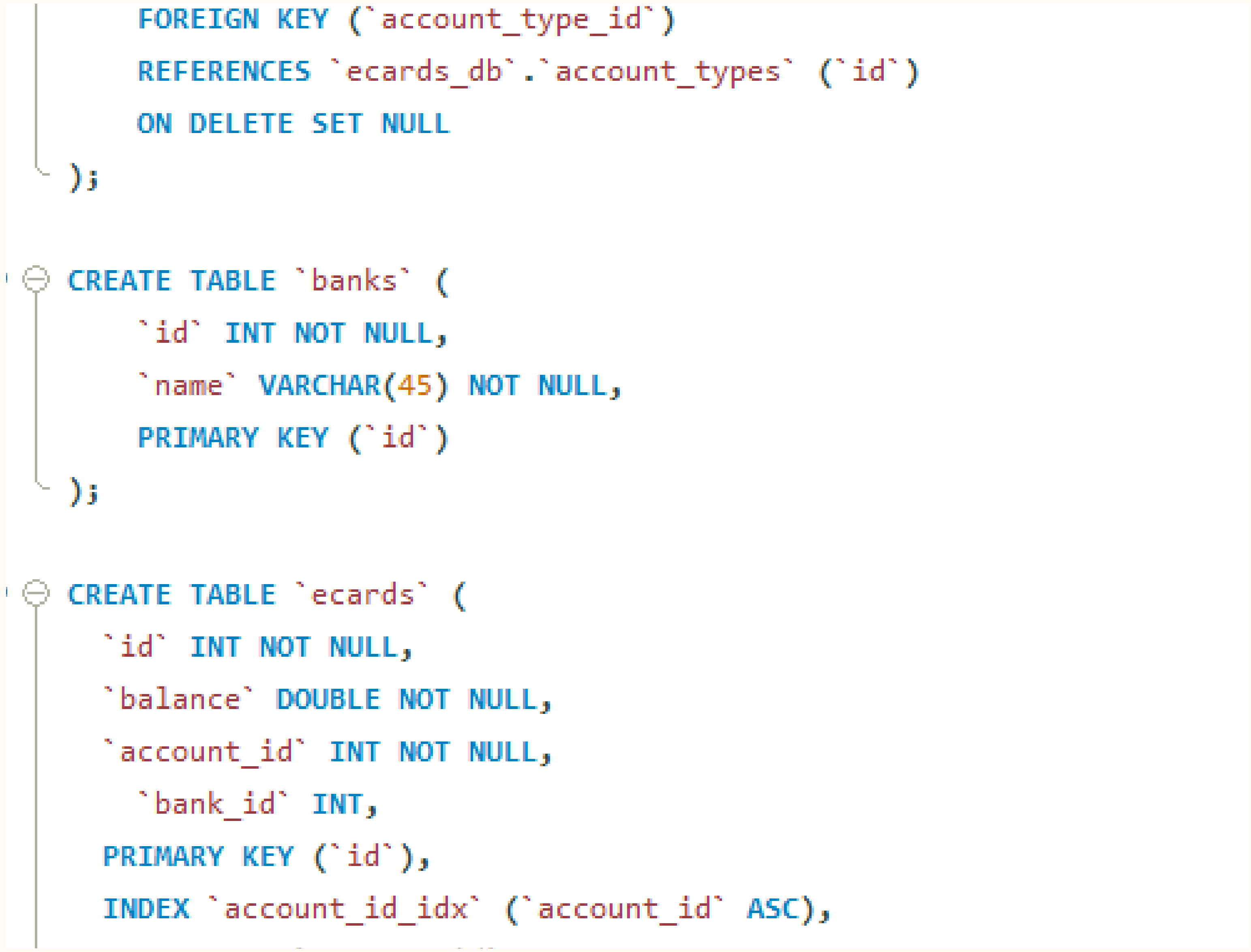


```
/* create the database */

CREATE DATABASE ecards_db;

/* use the database */
USE ecards_db;

/* create the tables */
CREATE TABLE `users` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `age` INT NOT NULL,
  PRIMARY KEY (`id`)
);
```

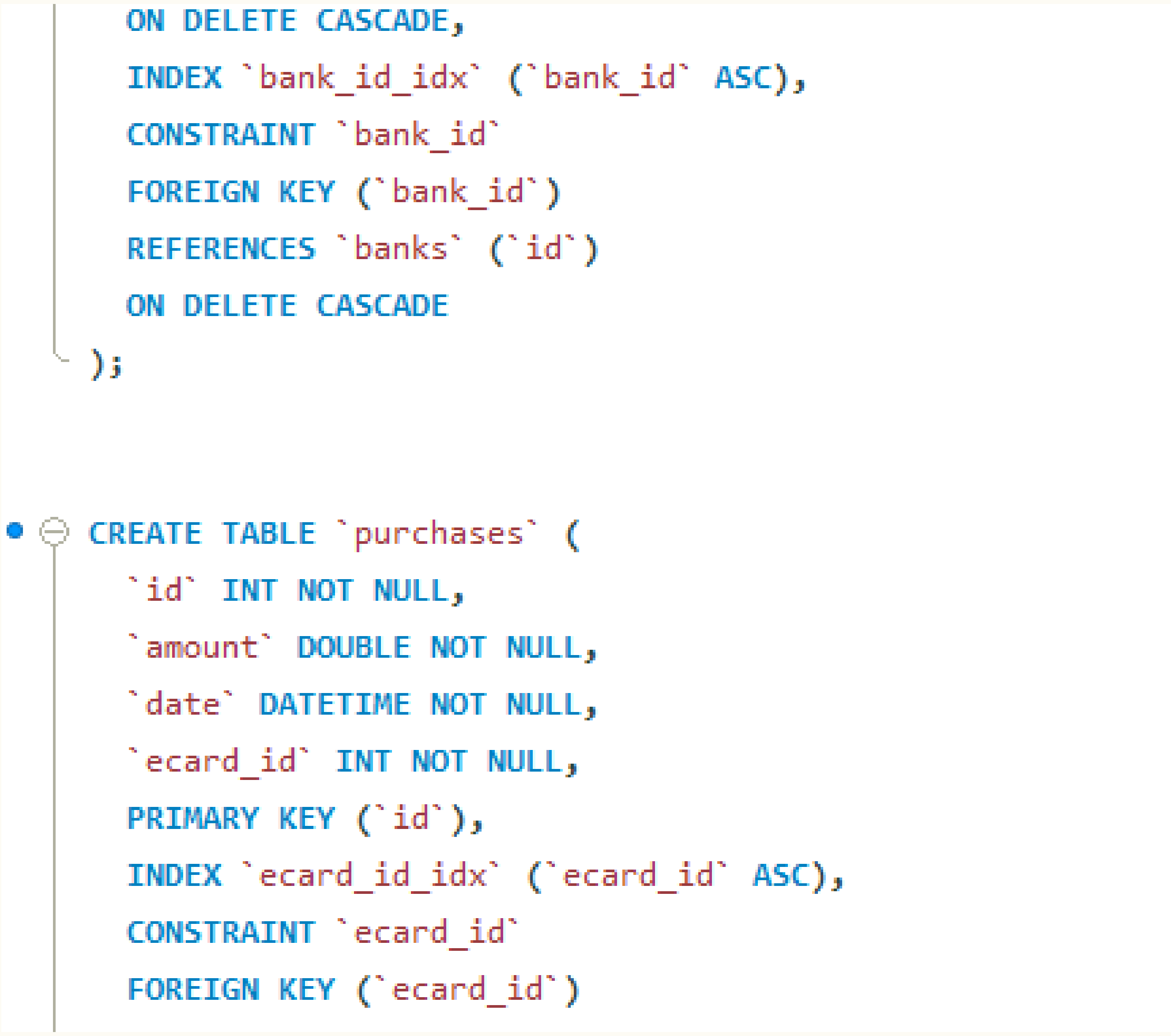


```
CREATE TABLE `account_types` (
    `id` INT NOT NULL AUTO_INCREMENT,
    `email` VARCHAR(45) NOT NULL,
    PRIMARY KEY (`id`)
);

CREATE TABLE `accounts` (
     `id` INT NOT NULL AUTO_INCREMENT,
     `email` VARCHAR(45) NOT NULL,
    `password` VARCHAR(45) NOT NULL,
    `user_id` INT,
    `account_type_id` INT,
    UNIQUE INDEX `email_UNIQUE` (`email` ASC),
    PRIMARY KEY (`id`),
    INDEX `user_id_idx` (`user_id` ASC) ,
    INDEX `account_type_id_idx` (`account_type_id` ASC),
    CONSTRAINT `user_id`
    FOREIGN KEY (`user_id`)
```

Database Schema

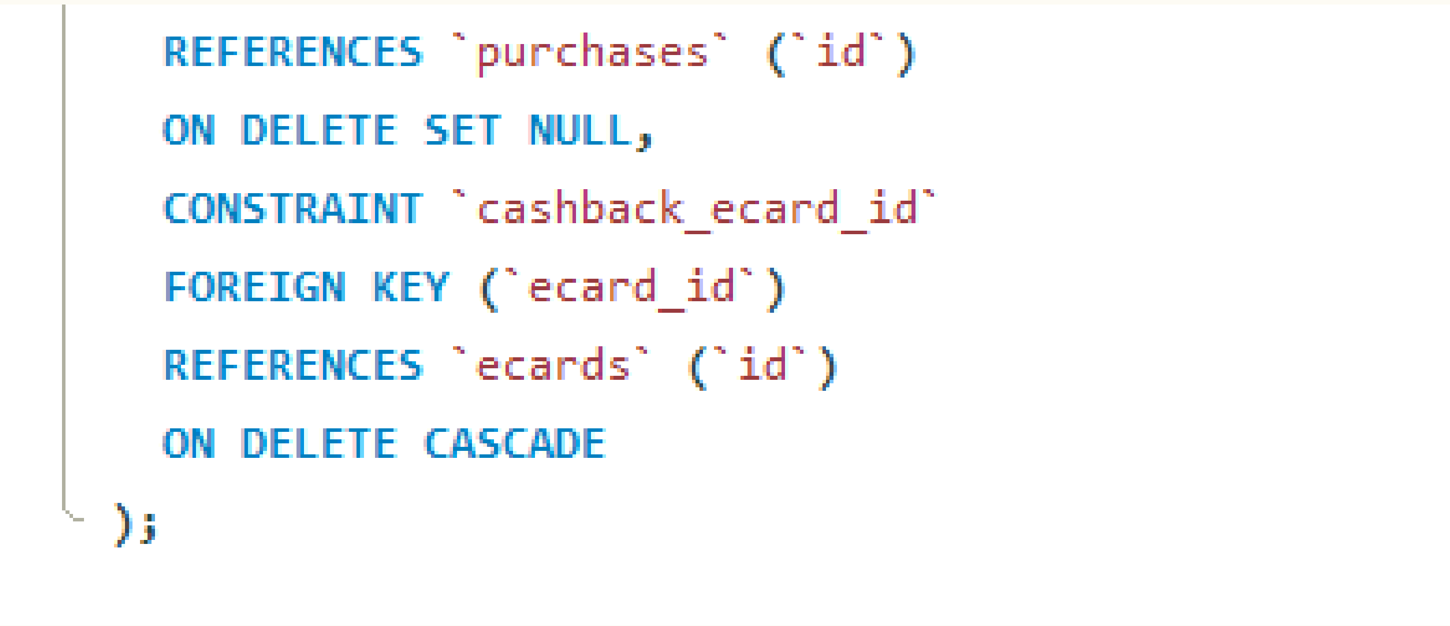


```
  FOREIGN KEY (`account_type_id`)
  REFERENCES `ecards_db`.`account_types` (`id`)
  ON DELETE SET NULL
);

CREATE TABLE `banks` (
  `id` INT NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE `ecards` (
 `id` INT NOT NULL,
 `balance` DOUBLE NOT NULL,
 `account_id` INT NOT NULL,
  `bank_id` INT,
 PRIMARY KEY (`id`),
 INDEX `account_id_idx` (`account_id` ASC),
```

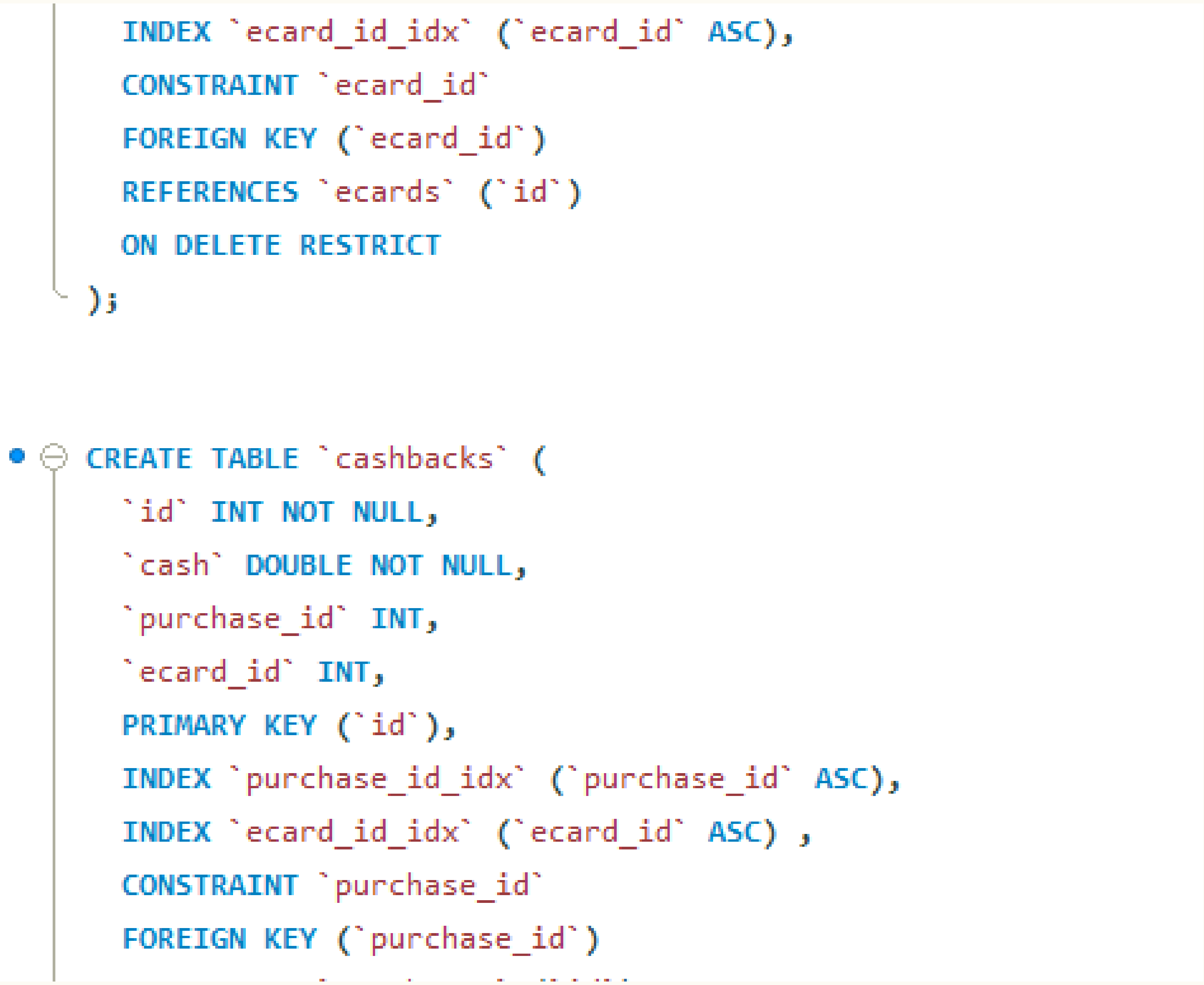


```
 ON DELETE CASCADE,
 INDEX `bank_id_idx` (`bank_id` ASC),
 CONSTRAINT `bank_id`
 FOREIGN KEY (`bank_id`)
 REFERENCES `banks` (`id`)
 ON DELETE CASCADE
);


CREATE TABLE `purchases` (
 `id` INT NOT NULL,
 `amount` DOUBLE NOT NULL,
 `date` DATETIME NOT NULL,
 `ecard_id` INT NOT NULL,
 PRIMARY KEY (`id`),
 INDEX `ecard_id_idx` (`ecard_id` ASC),
 CONSTRAINT `ecard_id`
 FOREIGN KEY (`ecard_id`)
```

Database Schema

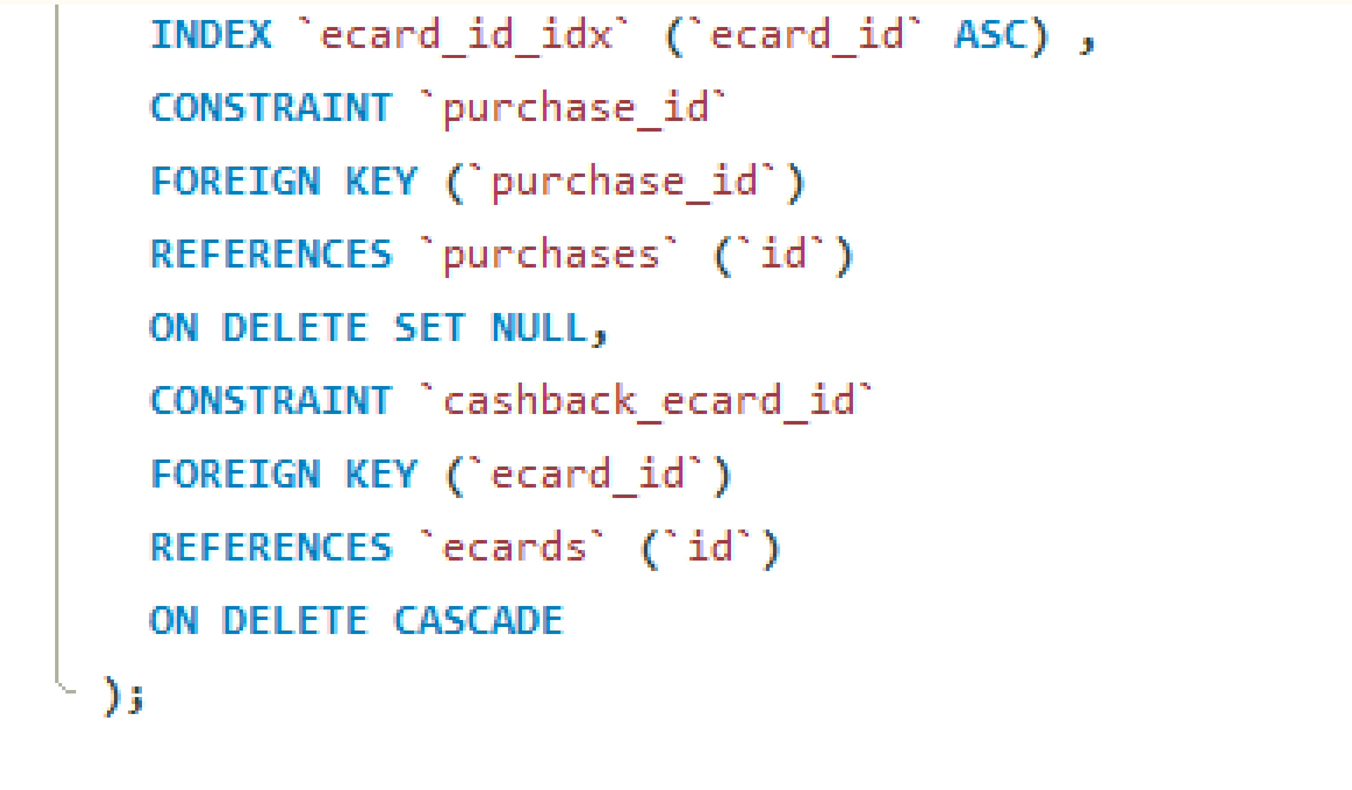


```
    REFERENCES `purchases` (`id`)
    ON DELETE SET NULL,
    CONSTRAINT `cashback_ecard_id`
    FOREIGN KEY (`ecard_id`)
    REFERENCES `ecards` (`id`)
    ON DELETE CASCADE
);
```

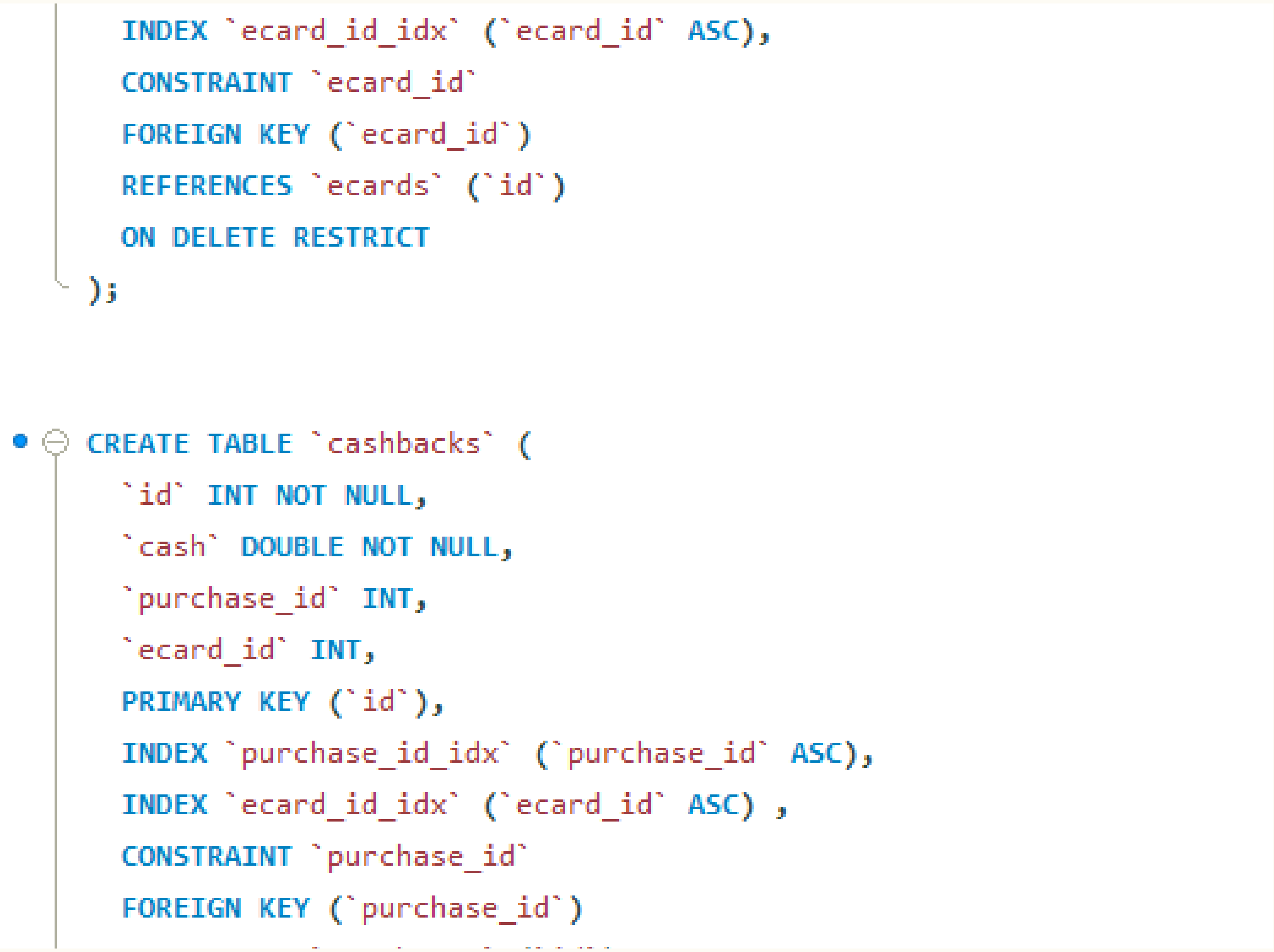


```
    INDEX `ecard_id_idx` (`ecard_id` ASC),
    CONSTRAINT `ecard_id`
    FOREIGN KEY (`ecard_id`)
    REFERENCES `ecards` (`id`)
    ON DELETE RESTRICT
);


CREATE TABLE `cashbacks` (
    `id` INT NOT NULL,
    `cash` DOUBLE NOT NULL,
    `purchase_id` INT,
    `ecard_id` INT,
    PRIMARY KEY (`id`),
    INDEX `purchase_id_idx` (`purchase_id` ASC),
    INDEX `ecard_id_idx` (`ecard_id` ASC) ,
    CONSTRAINT `purchase_id`
    FOREIGN KEY (`purchase_id`)
```

Database Schema



```
    INDEX `ecard_id_idx` (`ecard_id` ASC) ,
    CONSTRAINT `purchase_id`
    FOREIGN KEY (`purchase_id`)
    REFERENCES `purchases` (`id`)
    ON DELETE SET NULL,
    CONSTRAINT `cashback_ecard_id`
    FOREIGN KEY (`ecard_id`)
    REFERENCES `ecards` (`id`)
    ON DELETE CASCADE
);
```



```
    INDEX `ecard_id_idx` (`ecard_id` ASC),
    CONSTRAINT `ecard_id`
    FOREIGN KEY (`ecard_id`)
    REFERENCES `ecards` (`id`)
    ON DELETE RESTRICT
);


CREATE TABLE `cashbacks` (
    `id` INT NOT NULL,
    `cash` DOUBLE NOT NULL,
    `purchase_id` INT,
    `ecard_id` INT,
    PRIMARY KEY (`id`),
    INDEX `purchase_id_idx` (`purchase_id` ASC),
    INDEX `ecard_id_idx` (`ecard_id` ASC) ,
    CONSTRAINT `purchase_id`
    FOREIGN KEY (`purchase_id`)
```

Queries:

```
SELECT ec.id, SUM(p.amount)
AS total_purchase_amount
FROM ecards ec
JOIN purchases p ON ec.id = p.ecard_id
GROUP BY ec.id;

SELECT u.id, u.name, a.email
FROM users u
LEFT JOIN accounts a ON u.id = a.user_id;

SELECT a.id, a.email, a.user_id, at.name
FROM accounts a
CROSS JOIN account_types at;

SELECT * FROM users
NATURAL JOIN accounts;

SELECT *
FROM purchases
WHERE ecard_id IN (SELECT id FROM ecards WHERE balance > 100);
```

Queries:

```
SELECT * FROM users WHERE name LIKE "b%";

SELECT * FROM purchases WHERE amount > 50;

SELECT u.name AS user_name, a.email, e.balance
FROM users u
JOIN accounts a ON u.id = a.user_id
JOIN ecards e ON a.id = e.account_id;


SELECT id, amount, date, ecard_id, NULL AS cash, NULL AS purchase_id
FROM purchases
UNION ALL
SELECT id, NULL AS amount, NULL AS date, ecard_id, cash, purchase_id
FROM cashbacks;


SELECT p.ecard_id
FROM purchases p
WHERE p.ecard_id NOT IN (
 SELECT cb.ecard_id
 FROM cashbacks cb
);
```

Queries:

```
SELECT * FROM purchases
ORDER BY date DESC;

SELECT u.id, u.name, SUM(cb.cash) AS total_cashback
FROM users u
JOIN accounts a ON u.id = a.user_id
JOIN ecards e ON a.id = e.account_id
JOIN purchases p ON e.id = p.ecard_id
JOIN cashbacks cb ON p.id = cb.purchase_id
GROUP BY u.id, u.name;

SELECT DISTINCT name
FROM users;

SELECT u.name AS user_name, a.email AS account_email
FROM users u
JOIN accounts a ON u.id = a.user_id;
```

```
* create procedure */
DELIMITER $$
CREATE PROCEDURE CalculateTotalCashback (
   IN userID INT,
   OUT totalCashback DECIMAL(10, 2)
)
BEGIN
   SET totalCashback = 0;

   SELECT SUM(cb.cash) INTO totalCashback
   FROM users u
   JOIN accounts a ON u.id = a.user_id
   JOIN ecards e ON a.id = e.account_id
   JOIN purchases p ON e.id = p.ecard_id
   JOIN cashbacks cb ON p.id = cb.purchase_id
   WHERE u.id = userID;
END $$
DELIMITER ;
```

Queries:

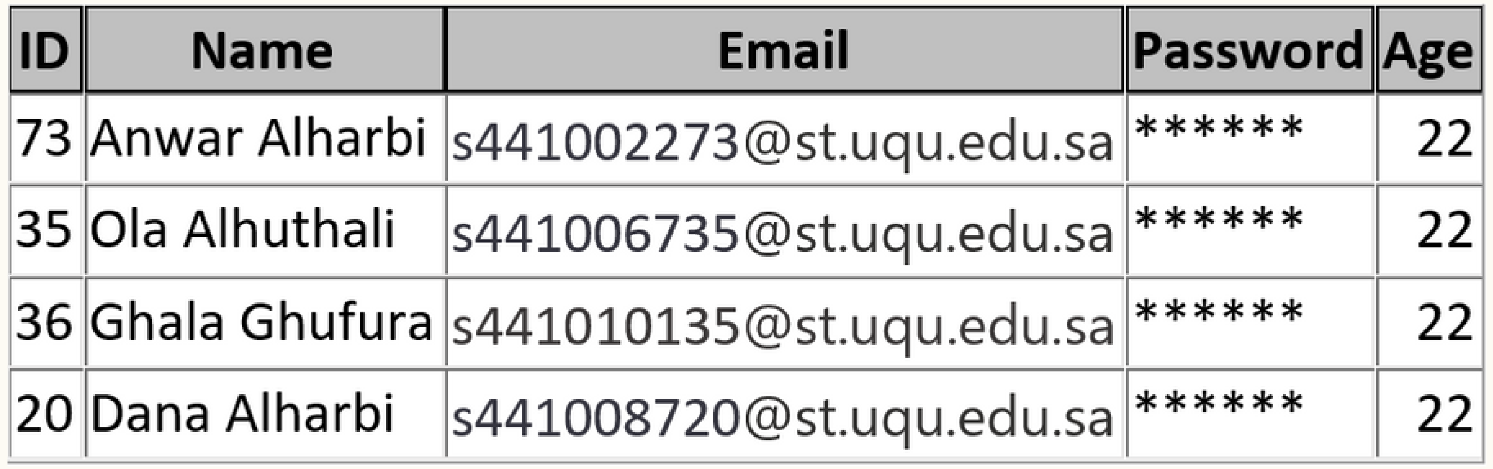
```
/* create function */
DELIMITER $$
CREATE FUNCTION CalculateAverageBalance (userId INT) RETURNS DECIMAL(10, 2)
DETERMINISTIC
BEGIN
    DECLARE avgBalance DECIMAL(10, 2);

    SELECT AVG(e.balance) INTO avgBalance
    FROM users u
    JOIN accounts a ON u.id = a.user_id
    JOIN ecards e ON a.id = e.account_id
    WHERE u.id = userId;

    RETURN avgBalance;
END $$
DELIMITER ;
```

Queries:

```
* create trigger */
DELIMITER $$
CREATE TRIGGER UpdateEcardBalance
AFTER INSERT ON purchases
FOR EACH ROW
BEGIN
   DECLARE ecardBalance DECIMAL(10, 2);
   SELECT balance INTO ecardBalance
   FROM ecards
   WHERE id = NEW.ecard_id;
   SET ecardBalance = ecardBalance - NEW.amount;
   UPDATE ecards
   SET balance = ecardBalance
   WHERE id = NEW.ecard_id;
END $$
DELIMITER ;
```

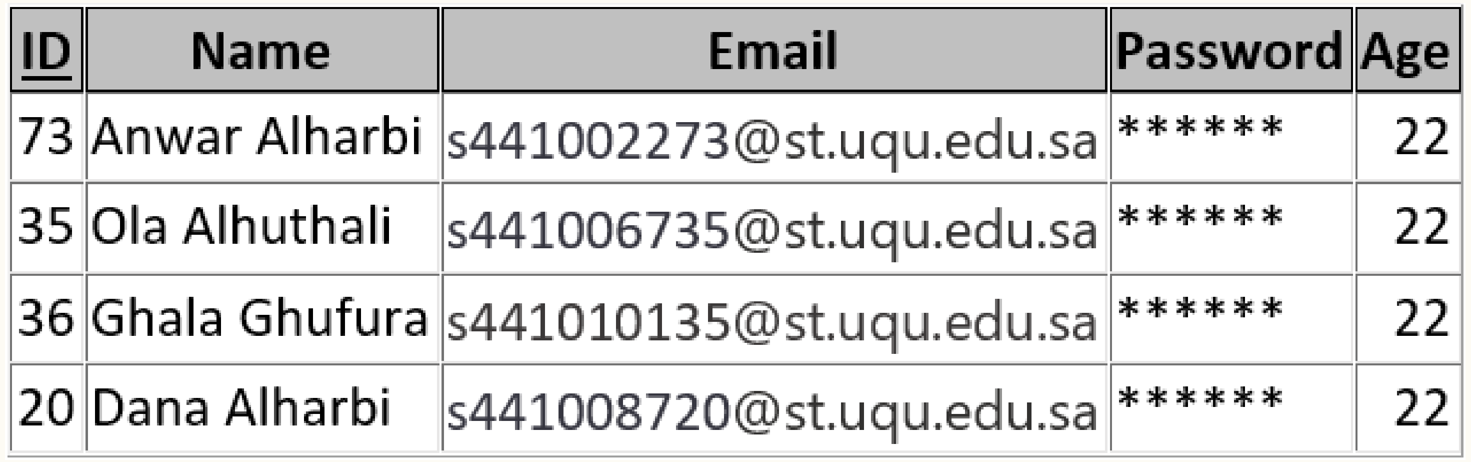
Normalization

Normalizing users table:



ID	Name	Email	Password	Age
73	Anwar Alharbi	s441002273@st.uqu.edu.sa	******	22
35	Ola Alhuthali	s441006735@st.uqu.edu.sa	******	22
36	Ghala Ghufura	s441010135@st.uqu.edu.sa	******	22
20	Dana Alharbi	s441008720@st.uqu.edu.sa	******	22

NF1

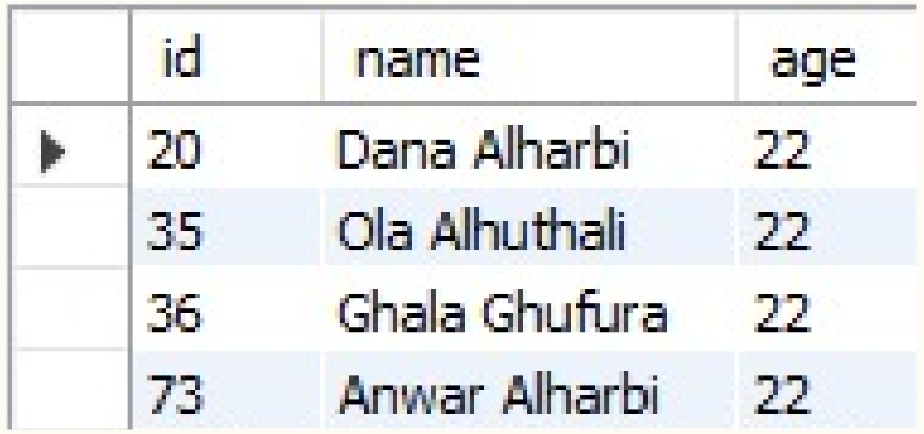


ID	Name	Email	Password	Age
73	Anwar Alharbi	s441002273@st.uqu.edu.sa	******	22
35	Ola Alhuthali	s441006735@st.uqu.edu.sa	******	22
36	Ghala Ghufura	s441010135@st.uqu.edu.sa	******	22
20	Dana Alharbi	s441008720@st.uqu.edu.sa	******	22

Normalization

NF2

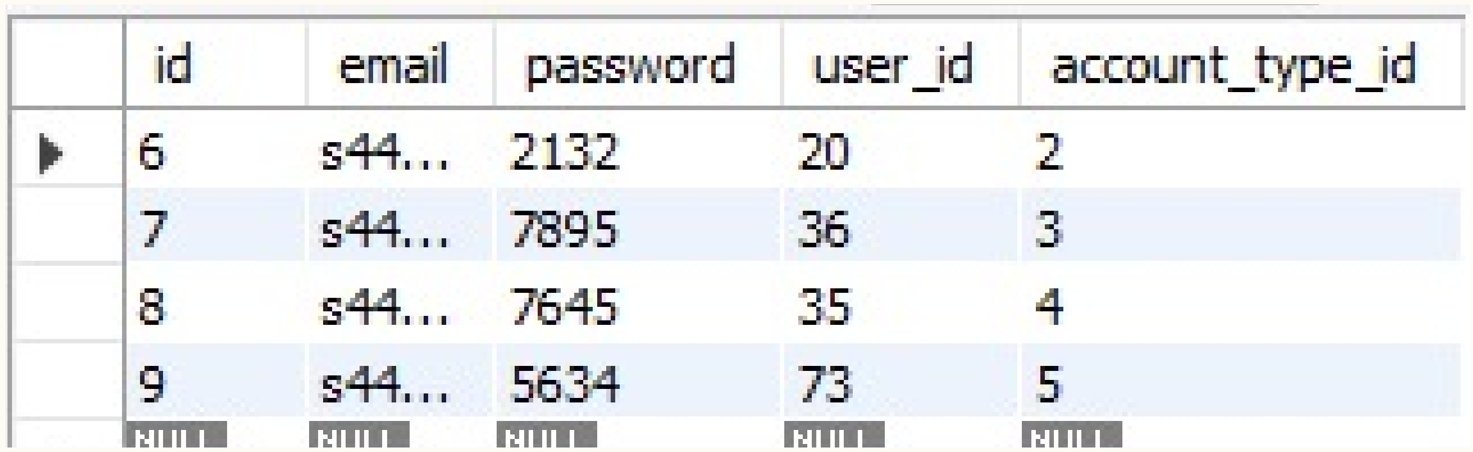
users table



	id	name	age
▶	20	Dana Alharbi	22
	35	Ola Alhuthali	22
	36	Ghala Ghufura	22
	73	Anwar Alharbi	22

NF3

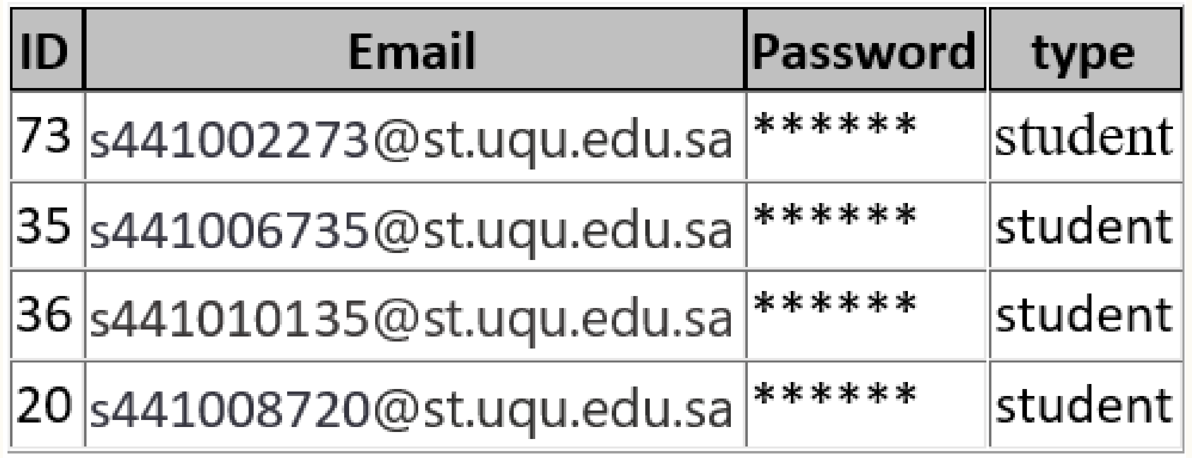
accounts table



	id	email	password	user_id	account_type_id
▶	6	s44...	2132	20	2
	7	s44...	7895	36	3
	8	s44...	7645	35	4
	9	s44...	5634	73	5
	NULL	NULL	NULL	NULL	NULL

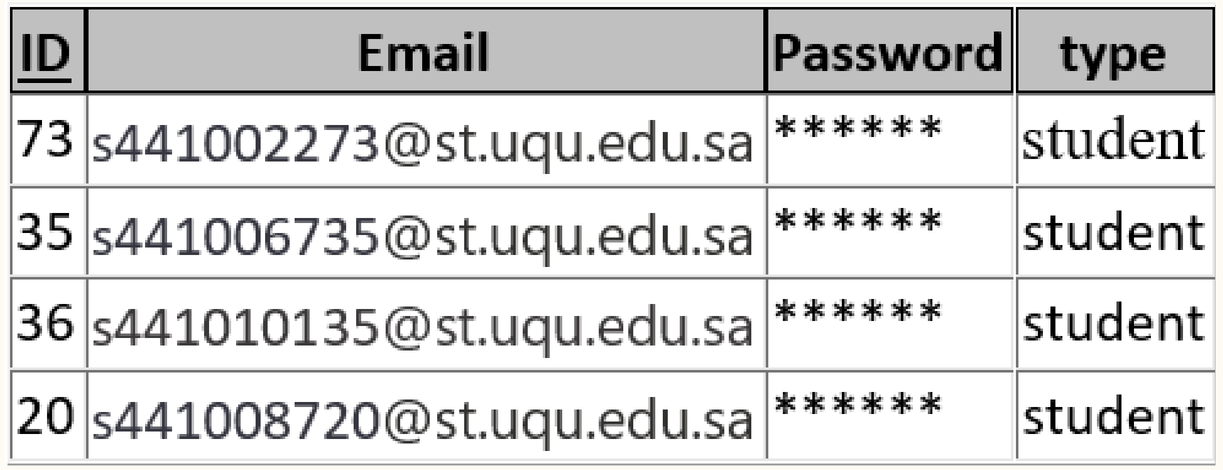
Normalization

Normalizing accounts table:



ID	Email	Password	type
73	s441002273@st.uqu.edu.sa	******	student
35	s441006735@st.uqu.edu.sa	******	student
36	s441010135@st.uqu.edu.sa	******	student
20	s441008720@st.uqu.edu.sa	******	student

NF1

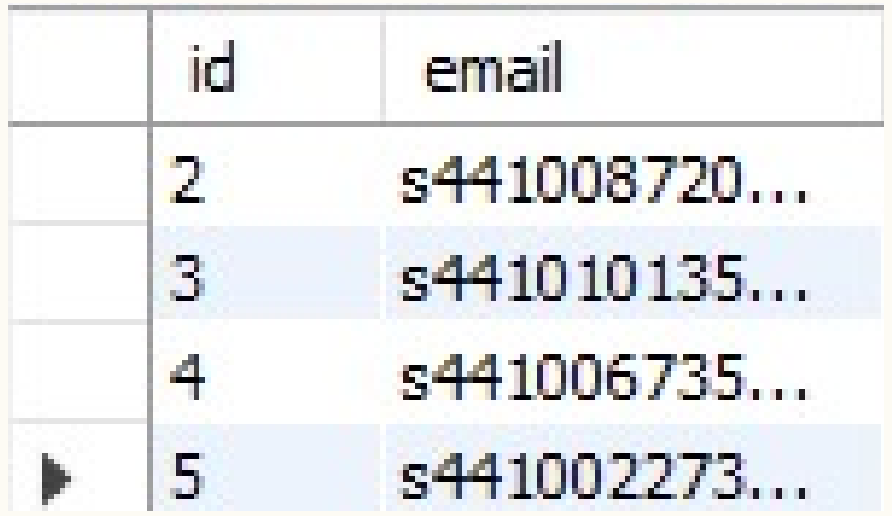


ID	Email	Password	type
73	s441002273@st.uqu.edu.sa	******	student
35	s441006735@st.uqu.edu.sa	******	student
36	s441010135@st.uqu.edu.sa	******	student
20	s441008720@st.uqu.edu.sa	******	student

Normalization

NF2

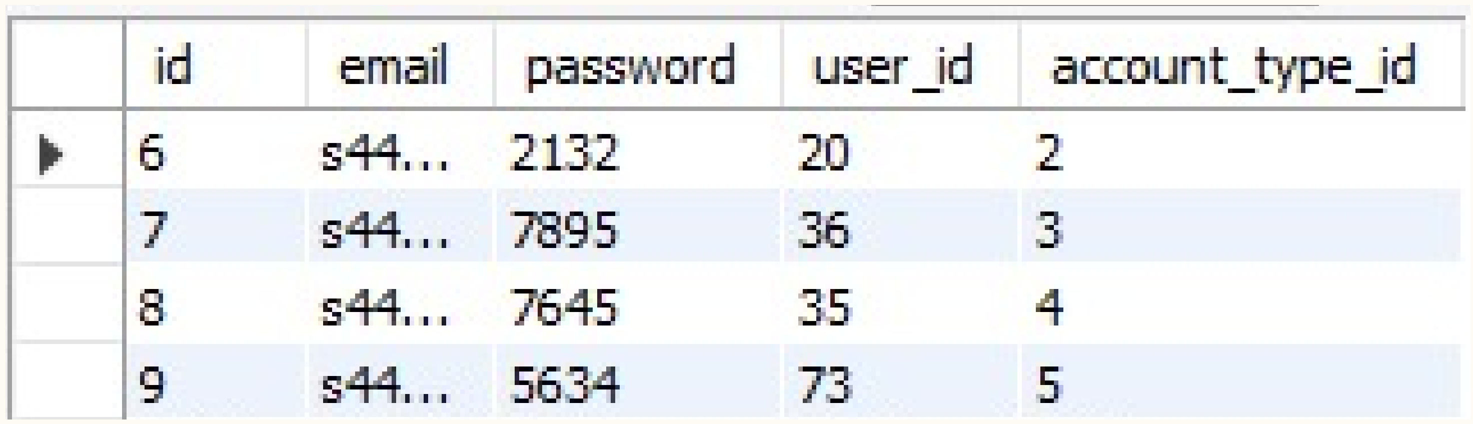
accounts types table



	id	email
	2	s441008720...
	3	s441010135...
	4	s441006735...
▶	5	s441002273...

NF2

accounts table



	id	email	password	user_id	account_type_id
▶	6	s44...	2132	20	2
	7	s44...	7895	36	3
	8	s44...	7645	35	4
	9	s44...	5634	73	5

Conclusion

Team members

Danah Mansour Alharbi
Ola Jameel Alhuthali
Anwar Yousef Alharbi
Ghala Yaser Ghufura