Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Базы данных (БД)

ОТЧЕТ

по лабораторной работе №7

Выполнил

студент: гр. 851006 Верещагин Н.В.

Проверил: Фадеева Е.Е.

Минск 2021

1. **Процедуры:**

Удалить все расписание до указанной даты:

CREATE PROCEDURE CLEAR\_SCHEDULE\_TO\_DATE(to\_date DATETIME)

BEGIN

    DELETE

        FROM

            schedule

        WHERE

            to\_date >= schedule.sch\_time\_end

END;

Событие для очистки истории каждый месяц:

CREATE EVENT CLEAR\_OLD\_SCHEDULE

    ON schedule

    EVERY 1 MONTH

    STARTS '2021-01-01 00:00:00'

    DO

    CALL CLEAR\_SCHEDULE\_TO\_DATE(CURRENT\_TIME());

Удалить всю историю для выбранного пользователя:

CREATE PROCEDURE CLEAR\_ALL\_HISTORY\_FOR\_USER(to\_user\_id INTEGER UNSIGNED)

BEGIN

    DELETE

        FROM

            edit\_history

        WHERE

            to\_user\_id = edit\_history.use\_id

END;

Удалить новости до указанной даты:

CREATE PROCEDURE CLEAR\_NEWS\_TO\_DATE(to\_date DATETIME)

BEGIN

    DELETE

        FROM

            news

        WHERE

            to\_date >= news.new\_time

END;

Установить дату для курса:

CREATE PROCEDURE SET\_DATES\_FOR\_COURSE(course\_id INTEGER UNSIGNED, date\_from DATE, date\_to DATE)

BEGIN

    UPDATE

        course

    SET

        course.cou\_date\_from = date\_from,

        course.cou\_date\_to = date\_to

    WHERE

        course.cou\_id = course\_id

END;

1. **Функции:**

Получить все пары в диапазоне времени:

CREATE FUNCTION GET\_ALL\_CLASSES\_BETWEEN\_TIME(from\_time DATETIME, to\_time DATETIME)

RETURN TABLE

AS

    RETURN

        SELECT

            class.\*

        FROM

            class

            INNER JOIN schedule ON class.cla\_id = schedule.cla\_id

        WHERE

            (from\_time BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end) OR

            (to\_time BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end)

Получить все пары в указанном месте:

CREATE FUNCTION GET\_ALL\_CLASSES\_IN\_PLACE(place\_name VARCHAR(350))

RETURN TABLE

AS

    RETURN

        SELECT

            class.\*

        FROM

            class

            INNER JOIN schedule ON class.cla\_id = schedule.cla\_id

            INNER JOIN place ON schedule.pla\_id = place.pla\_id

        WHERE

            place\_name = place.pla\_name

Получить все пары для преподавателя:

CREATE FUNCTION GET\_ALL\_CLASSES\_FOR\_TEACHER(teacher\_id INTEGER UNSIGNED)

RETURN TABLE

AS

    RETURN

        SELECT

            \*

        FROM

            class

        WHERE

            teacher\_id = class.cla\_teacher\_id

Получить оценку за задачу:

CREATE FUNCTION GET\_MARK(user\_id INTEGER UNSIGNED, task\_id INTEGER UNSIGNED)

RETURNS SMALLINT UNSIGNED

DETERMINISTIC

BEGIN

    RETURN

        SELECT

            gra\_mark

        FROM

            gradebook

        WHERE

            user\_id = gradebook.use\_id AND task\_id = gradebook.tas\_id

END;

Добавить новый курс:

CREATE FUNCTION ADD\_NEW\_COURSE(level\_id SMALLINT UNSIGNED, name\_id SMALLINT UNSIGNED, is\_individual BOOL)

RETURNS INTEGER UNSIGNED

DETERMINISTIC

BEGIN

    RETURN

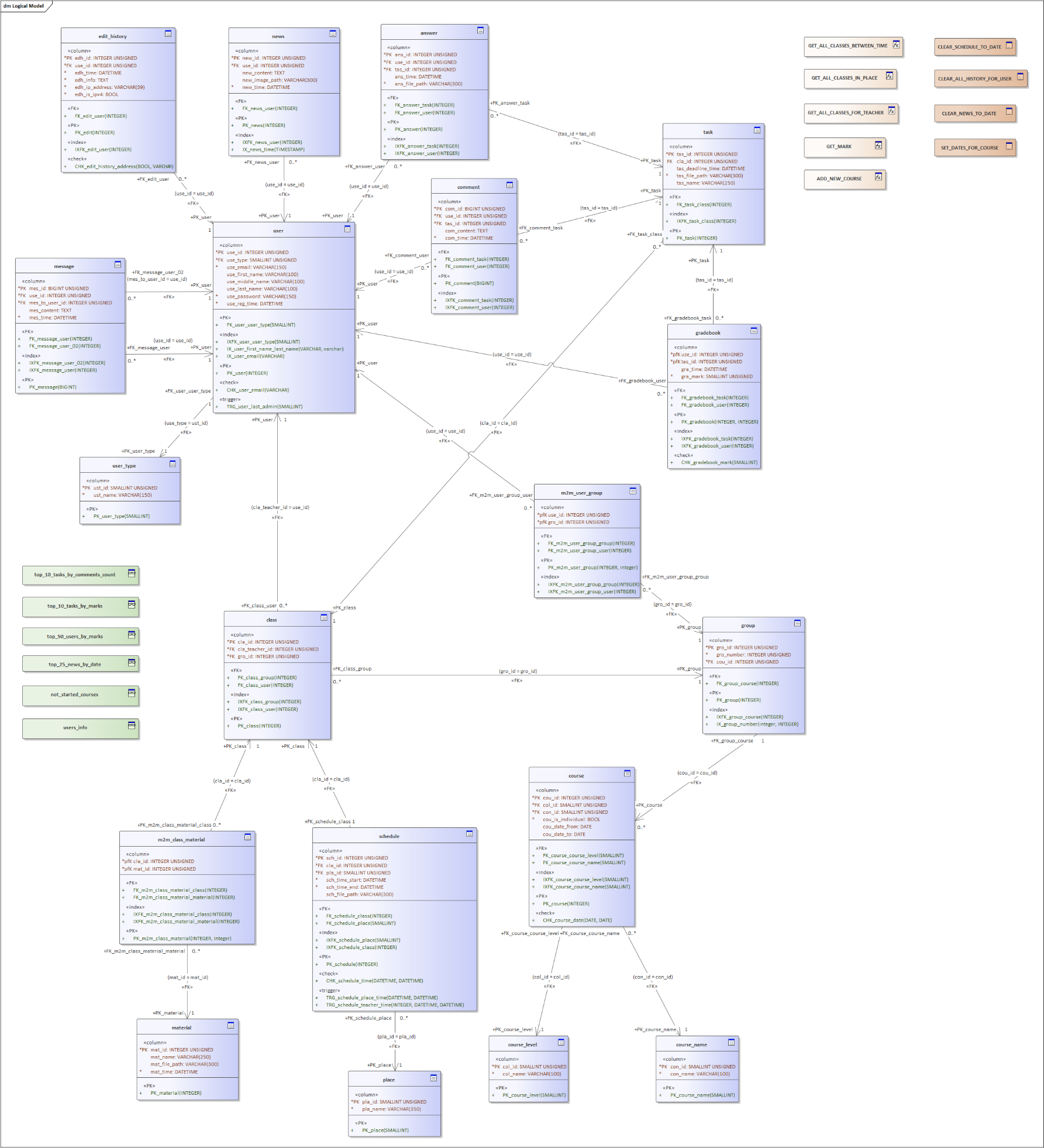
        INSERT INTO

            course (col\_id, con\_id, cou\_is\_individual)

            OUTPUT INSERTED.cou\_id

        VALUES (level\_id, name\_id, is\_individual);

END;



Код:

/\* ---------------------------------------------------- \*/

/\*  Generated by Enterprise Architect Version 15.0      \*/

/\*  Created On : 26-Apr-2021 6:08:35 PM                 \*/

/\*  DBMS       : MySql                      \*/

/\* ---------------------------------------------------- \*/

SET FOREIGN\_KEY\_CHECKS=0

;

/\* Drop Views \*/

DROP VIEW IF EXISTS `not\_started\_courses` CASCADE

;

DROP VIEW IF EXISTS `top\_10\_tasks\_by\_comments\_count` CASCADE

;

DROP VIEW IF EXISTS `top\_10\_tasks\_by\_marks` CASCADE

;

DROP VIEW IF EXISTS `top\_25\_news\_by\_date` CASCADE

;

DROP VIEW IF EXISTS `top\_50\_users\_by\_marks` CASCADE

;

DROP VIEW IF EXISTS `users\_info` CASCADE

;

/\* Drop Tables \*/

DROP TABLE IF EXISTS `answer` CASCADE

;

DROP TABLE IF EXISTS `class` CASCADE

;

DROP TABLE IF EXISTS `comment` CASCADE

;

DROP TABLE IF EXISTS `course` CASCADE

;

DROP TABLE IF EXISTS `course\_level` CASCADE

;

DROP TABLE IF EXISTS `course\_name` CASCADE

;

DROP TABLE IF EXISTS `edit\_history` CASCADE

;

DROP TABLE IF EXISTS `gradebook` CASCADE

;

DROP TABLE IF EXISTS `group` CASCADE

;

DROP TABLE IF EXISTS `m2m\_class\_material` CASCADE

;

DROP TABLE IF EXISTS `m2m\_user\_group` CASCADE

;

DROP TABLE IF EXISTS `material` CASCADE

;

DROP TABLE IF EXISTS `message` CASCADE

;

DROP TABLE IF EXISTS `news` CASCADE

;

DROP TABLE IF EXISTS `place` CASCADE

;

DROP TABLE IF EXISTS `schedule` CASCADE

;

DROP TABLE IF EXISTS `task` CASCADE

;

DROP TABLE IF EXISTS `user` CASCADE

;

DROP TABLE IF EXISTS `user\_type` CASCADE

;

/\* Drop Functions \*/

DROP FUNCTION IF EXISTS `ADD\_NEW\_COURSE`

;

DROP FUNCTION IF EXISTS `GET\_ALL\_CLASSES\_BETWEEN\_TIME`

;

DROP FUNCTION IF EXISTS `GET\_ALL\_CLASSES\_FOR\_TEACHER`

;

DROP FUNCTION IF EXISTS `GET\_ALL\_CLASSES\_IN\_PLACE`

;

DROP FUNCTION IF EXISTS `GET\_MARK`

;

/\* Create Functions \*/

DELIMITER //

CREATE FUNCTION ADD\_NEW\_COURSE(level\_id SMALLINT UNSIGNED, name\_id SMALLINT UNSIGNED, is\_individual BOOL)

RETURNS INTEGER UNSIGNED

DETERMINISTIC

BEGIN

    RETURN

        INSERT INTO

            course (col\_id, con\_id, cou\_is\_individual)

            OUTPUT INSERTED.cou\_id

        VALUES (level\_id, name\_id, is\_individual);

END;

//

DELIMITER ;

;

DELIMITER //

CREATE FUNCTION GET\_ALL\_CLASSES\_BETWEEN\_TIME(from\_time DATETIME, to\_time DATETIME)

RETURN TABLE

AS

    RETURN

        SELECT

            class.\*

        FROM

            class

            INNER JOIN schedule ON class.cla\_id = schedule.cla\_id

        WHERE

            (from\_time BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end) OR

            (to\_time BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end)

//

DELIMITER ;

;

DELIMITER //

CREATE FUNCTION GET\_ALL\_CLASSES\_FOR\_TEACHER(teacher\_id INTEGER UNSIGNED)

RETURN TABLE

AS

    RETURN

        SELECT

            \*

        FROM

            class

        WHERE

            teacher\_id = class.cla\_teacher\_id

//

DELIMITER ;

;

DELIMITER //

CREATE FUNCTION GET\_ALL\_CLASSES\_IN\_PLACE(place\_name VARCHAR(350))

RETURN TABLE

AS

    RETURN

        SELECT

            class.\*

        FROM

            class

            INNER JOIN schedule ON class.cla\_id = schedule.cla\_id

            INNER JOIN place ON schedule.pla\_id = place.pla\_id

        WHERE

            place\_name = place.pla\_name

//

DELIMITER ;

;

DELIMITER //

CREATE FUNCTION GET\_MARK(user\_id INTEGER UNSIGNED, task\_id INTEGER UNSIGNED)

RETURNS SMALLINT UNSIGNED

DETERMINISTIC

BEGIN

    RETURN

        SELECT

            gra\_mark

        FROM

            gradebook

        WHERE

            user\_id = gradebook.use\_id AND task\_id = gradebook.tas\_id

END;

//

DELIMITER ;

;

/\* Create Tables \*/

CREATE TABLE `answer`

(

    `ans\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_id` INT UNSIGNED NOT NULL,

    `tas\_id` INT UNSIGNED NOT NULL,

    `ans\_time` DATETIME NULL,

    `ans\_file\_path` VARCHAR(300) NOT NULL COMMENT 'The path to the file with the answer',

    CONSTRAINT `PK\_answer` PRIMARY KEY (`ans\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `class`

(

    `cla\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `cla\_teacher\_id` INT UNSIGNED NOT NULL,

    `gro\_id` INT UNSIGNED NOT NULL,

    CONSTRAINT `PK\_class` PRIMARY KEY (`cla\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `comment`

(

    `com\_id` BIGINT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_id` INT UNSIGNED NOT NULL,

    `tas\_id` INT UNSIGNED NOT NULL,

    `com\_content` TEXT NULL,

    `com\_time` DATETIME NOT NULL,

    CONSTRAINT `PK\_comment` PRIMARY KEY (`com\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `course`

(

    `cou\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `col\_id` SMALLINT UNSIGNED NOT NULL,

    `con\_id` SMALLINT UNSIGNED NOT NULL,

    `cou\_is\_individual` BOOL NOT NULL COMMENT 'Flag that indicates whether the course is individual or group',

    `cou\_date\_from` DATE NULL,

    `cou\_date\_to` DATE NULL,

    CONSTRAINT `PK\_course` PRIMARY KEY (`cou\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `course\_level`

(

    `col\_id` SMALLINT UNSIGNED NOT NULL,

    `col\_name` VARCHAR(100) NOT NULL,

    CONSTRAINT `PK\_course\_level` PRIMARY KEY (`col\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `course\_name`

(

    `con\_id` SMALLINT UNSIGNED NOT NULL,

    `con\_name` VARCHAR(100) NOT NULL,

    CONSTRAINT `PK\_course\_name` PRIMARY KEY (`con\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `edit\_history`

(

    `edh\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_id` INT UNSIGNED NOT NULL,

    `edh\_time` DATETIME NOT NULL,

    `edh\_info` TEXT NOT NULL COMMENT 'Change information is stored in JSON format',

    `edh\_ip\_address` VARCHAR(39) NOT NULL COMMENT 'It stores either an ipv4 address or an ipv6 address',

    `edh\_is\_ipv4` BOOL NOT NULL COMMENT 'A flag that indicates whether the address is stored in ipv4 format or in ipv6 format',

    CONSTRAINT `PK\_edit` PRIMARY KEY (`edh\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `gradebook`

(

    `use\_id` INT UNSIGNED NOT NULL,

    `tas\_id` INT UNSIGNED NOT NULL,

    `gra\_time` DATETIME NULL,

    `gra\_mark` SMALLINT UNSIGNED NOT NULL,

    CONSTRAINT `PK\_gradebook` PRIMARY KEY (`use\_id` ASC, `tas\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `group`

(

    `gro\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `gro\_number` INT UNSIGNED NOT NULL,

    `cou\_id` INT UNSIGNED NOT NULL,

    CONSTRAINT `PK\_group` PRIMARY KEY (`gro\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `m2m\_class\_material`

(

    `cla\_id` INT UNSIGNED NOT NULL,

    `mat\_id` INT UNSIGNED NOT NULL,

    CONSTRAINT `PK\_m2m\_class\_material` PRIMARY KEY (`cla\_id` ASC, `mat\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `m2m\_user\_group`

(

    `use\_id` INT UNSIGNED NOT NULL,

    `gro\_id` INT UNSIGNED NOT NULL,

    CONSTRAINT `PK\_m2m\_user\_group` PRIMARY KEY (`gro\_id` ASC, `use\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `material`

(

    `mat\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `mat\_name` VARCHAR(250) NULL,

    `mat\_file\_path` VARCHAR(300) NULL COMMENT 'File path with material',

    `mat\_time` DATETIME NOT NULL,

    CONSTRAINT `PK\_material` PRIMARY KEY (`mat\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `message`

(

    `mes\_id` BIGINT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_id` INT UNSIGNED NOT NULL,

    `mes\_to\_user\_id` INT UNSIGNED NOT NULL COMMENT 'The id of the user to whom the message is intended is stored here',

    `mes\_content` TEXT NULL,

    `mes\_time` DATETIME NOT NULL,

    CONSTRAINT `PK\_message` PRIMARY KEY (`mes\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `news`

(

    `new\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_id` INT UNSIGNED NOT NULL,

    `new\_content` TEXT NULL,

    `new\_image\_path` VARCHAR(300) NULL COMMENT 'The path to the picture',

    `new\_time` DATETIME NOT NULL,

    CONSTRAINT `PK\_news` PRIMARY KEY (`new\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `place`

(

    `pla\_id` SMALLINT UNSIGNED NOT NULL,

    `pla\_name` VARCHAR(350) NOT NULL COMMENT 'Name of the venue',

    CONSTRAINT `PK\_place` PRIMARY KEY (`pla\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `schedule`

(

    `sch\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `cla\_id` INT UNSIGNED NOT NULL,

    `pla\_id` SMALLINT UNSIGNED NOT NULL,

    `sch\_time\_start` DATETIME NOT NULL,

    `sch\_time\_end` DATETIME NOT NULL,

    `sch\_file\_path` VARCHAR(300) NULL COMMENT 'Path to a file with additional information for the lesson',

    CONSTRAINT `PK\_schedule` PRIMARY KEY (`sch\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `task`

(

    `tas\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `cla\_id` INT UNSIGNED NULL,

    `tas\_deadline\_time` DATETIME NULL COMMENT 'Time until which you can send a response to the assignment',

    `tas\_file\_path` VARCHAR(300) NOT NULL COMMENT 'The path to the task file',

    `tas\_name` VARCHAR(250) NULL,

    CONSTRAINT `PK\_task` PRIMARY KEY (`tas\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `user`

(

    `use\_id` INT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `use\_type` SMALLINT UNSIGNED NOT NULL,

    `use\_email` VARCHAR(150) NOT NULL,

    `use\_first\_name` VARCHAR(100) NULL,

    `use\_middle\_name` VARCHAR(100) NULL,

    `use\_last\_name` VARCHAR(100) NULL,

    `use\_password` VARCHAR(150) NOT NULL,

    `use\_reg\_time` DATETIME NOT NULL,

    CONSTRAINT `PK\_user` PRIMARY KEY (`use\_id` ASC)

)

COLLATE utf8\_general\_ci

;

CREATE TABLE `user\_type`

(

    `ust\_id` SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

    `ust\_name` VARCHAR(150) NOT NULL,

    CONSTRAINT `PK\_user\_type` PRIMARY KEY (`ust\_id` ASC)

)

COLLATE utf8\_general\_ci

;

/\* Create Primary Keys, Indexes, Uniques, Checks \*/

ALTER TABLE `answer`

 ADD INDEX `IXFK\_answer\_task` (`tas\_id` ASC)

;

ALTER TABLE `answer`

 ADD INDEX `IXFK\_answer\_user` (`use\_id` ASC)

;

ALTER TABLE `class`

 ADD INDEX `IXFK\_class\_group` (`gro\_id` ASC)

;

ALTER TABLE `class`

 ADD INDEX `IXFK\_class\_user` (`cla\_teacher\_id` ASC)

;

ALTER TABLE `comment`

 ADD INDEX `IXFK\_comment\_task` (`tas\_id` ASC)

;

ALTER TABLE `comment`

 ADD INDEX `IXFK\_comment\_user` (`use\_id` ASC)

;

ALTER TABLE `course`

 ADD CONSTRAINT `CHK\_course\_date` CHECK (cou\_date\_from <= cou\_date\_to)

;

ALTER TABLE `course`

 ADD INDEX `IXFK\_course\_course\_level` (`col\_id` ASC)

;

ALTER TABLE `course`

 ADD INDEX `IXFK\_course\_course\_name` (`con\_id` ASC)

;

ALTER TABLE `edit\_history`

 ADD CONSTRAINT `CHK\_edit\_history\_address` CHECK ((edh\_is\_ipv4 AND IS\_IPV4(edh\_ip\_address)) OR

(NOT edh\_is\_ipv4 AND IS\_IPV6(edh\_ip\_address))

)

;

ALTER TABLE `edit\_history`

 ADD INDEX `IXFK\_edit\_user` (`use\_id` ASC)

;

ALTER TABLE `gradebook`

 ADD CONSTRAINT `CHK\_gradebook\_mark` CHECK (gra\_mark <= 10)

;

ALTER TABLE `gradebook`

 ADD INDEX `IXFK\_gradebook\_task` (`tas\_id` ASC)

;

ALTER TABLE `gradebook`

 ADD INDEX `IXFK\_gradebook\_user` (`use\_id` ASC)

;

ALTER TABLE `group`

 ADD INDEX `IXFK\_group\_course` (`cou\_id` ASC)

;

ALTER TABLE `group`

 ADD INDEX `IX\_group\_number` (`gro\_number` ASC, `cou\_id` ASC)

;

ALTER TABLE `m2m\_class\_material`

 ADD INDEX `IXFK\_m2m\_class\_material\_class` (`cla\_id` ASC)

;

ALTER TABLE `m2m\_class\_material`

 ADD INDEX `IXFK\_m2m\_class\_material\_material` (`mat\_id` ASC)

;

ALTER TABLE `m2m\_user\_group`

 ADD INDEX `IXFK\_m2m\_user\_group\_group` (`gro\_id` ASC)

;

ALTER TABLE `m2m\_user\_group`

 ADD INDEX `IXFK\_m2m\_user\_group\_user` (`use\_id` ASC)

;

ALTER TABLE `message`

 ADD INDEX `IXFK\_message\_user\_02` (`mes\_to\_user\_id` ASC)

;

ALTER TABLE `message`

 ADD INDEX `IXFK\_message\_user` (`use\_id` ASC)

;

ALTER TABLE `news`

 ADD INDEX `IXFK\_news\_user` (`use\_id` ASC)

;

ALTER TABLE `news`

 ADD INDEX `IX\_news\_time` (`new\_time` ASC)

;

ALTER TABLE `schedule`

 ADD CONSTRAINT `CHK\_schedule\_time` CHECK (sch\_time\_start <= sch\_time\_end)

;

ALTER TABLE `schedule`

 ADD INDEX `IXFK\_schedule\_place` (`pla\_id` ASC)

;

ALTER TABLE `schedule`

 ADD INDEX `IXFK\_schedule\_class` (`cla\_id` ASC)

;

DELIMITER //

CREATE TRIGGER 'TRG\_schedule\_place\_time' BEFORE INSERT ON schedule FOR EACH ROW

BEGIN

    IF EXISTS (

        SELECT

            \*

        FROM

            schedule

        WHERE

            schedule.pla\_id = NEW.pla\_id AND

            (

                NEW.sch\_time\_start BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end

            ) OR (

                NEW.sch\_time\_end BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end

            )

    )

    THEN

        SIGNAL SQLSTATE '45002'

        SET MESSAGE\_TEXT = 'You cannot schedule another activity during this time.'

        MYSQL\_ERRNO = 1002

    END IF;

END;

//

DELIMITER ;

;

DELIMITER //

CREATE TRIGGER 'TRG\_schedule\_teacher\_time' BEFORE INSERT ON schedule FOR EACH ROW

BEGIN

    IF EXISTS (

        SELECT

            \*

        FROM

            schedule

            INNER JOIN class ON class.cla\_id = schedule.cla\_id

        WHERE

            class.cla\_teacher\_id = NEW.cla\_teacher\_id AND

            (

                NEW.sch\_time\_start BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end

            ) OR (

                NEW.sch\_time\_end BETWEEN schedule.sch\_time\_start AND schedule.sch\_time\_end

            )

    )

    THEN

        SIGNAL SQLSTATE '45001'

        SET MESSAGE\_TEXT = 'You cannot assign several lessons to the same instructor at the same time.'

        MYSQL\_ERRNO = 1001

    END IF;

END;

//

DELIMITER ;

;

ALTER TABLE `task`

 ADD INDEX `IXFK\_task\_class` (`cla\_id` ASC)

;

ALTER TABLE `user`

 ADD CONSTRAINT `CHK\_user\_email` CHECK (use\_email LIKE '%\_@\_\_%.\_\_%')

;

ALTER TABLE `user`

 ADD INDEX `IXFK\_user\_user\_type` (`use\_type` ASC)

;

ALTER TABLE `user`

 ADD INDEX `IX\_user\_first\_name\_last\_name` (`use\_first\_name` ASC, `use\_last\_name` ASC)

;

ALTER TABLE `user`

 ADD INDEX `IX\_user\_email` (`use\_email` ASC)

;

DELIMITER //

CREATE TRIGGER 'TRG\_user\_last\_admin' BEFORE DELETE ON user FOR EACH ROW

BEGIN

    IF (

        SELECT

            \*

        FROM

            user\_type

            INNER JOIN user ON user.ust\_type = user\_type.ust\_id

        WHERE

            (user\_type.ust\_name = 'admin') AND

            (user.ust\_type = user\_type.ust\_id)

        < 2

    )

    THEN

        SIGNAL SQLSTATE '45003'

        SET MESSAGE\_TEXT = 'You cannot remove the last admin.'

        MYSQL\_ERRNO = 1003

    END IF;

END;

//

DELIMITER ;

;

/\* Create Foreign Key Constraints \*/

ALTER TABLE `answer`

 ADD CONSTRAINT `FK\_answer\_task`

    FOREIGN KEY (`tas\_id`) REFERENCES `task` (`tas\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `answer`

 ADD CONSTRAINT `FK\_answer\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `class`

 ADD CONSTRAINT `FK\_class\_group`

    FOREIGN KEY (`gro\_id`) REFERENCES `group` (`gro\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `class`

 ADD CONSTRAINT `FK\_class\_user`

    FOREIGN KEY (`cla\_teacher\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `comment`

 ADD CONSTRAINT `FK\_comment\_task`

    FOREIGN KEY (`tas\_id`) REFERENCES `task` (`tas\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `comment`

 ADD CONSTRAINT `FK\_comment\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `course`

 ADD CONSTRAINT `FK\_course\_course\_level`

    FOREIGN KEY (`col\_id`) REFERENCES `course\_level` (`col\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `course`

 ADD CONSTRAINT `FK\_course\_course\_name`

    FOREIGN KEY (`con\_id`) REFERENCES `course\_name` (`con\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `edit\_history`

 ADD CONSTRAINT `FK\_edit\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `gradebook`

 ADD CONSTRAINT `FK\_gradebook\_task`

    FOREIGN KEY (`tas\_id`) REFERENCES `task` (`tas\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `gradebook`

 ADD CONSTRAINT `FK\_gradebook\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `group`

 ADD CONSTRAINT `FK\_group\_course`

    FOREIGN KEY (`cou\_id`) REFERENCES `course` (`cou\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `m2m\_class\_material`

 ADD CONSTRAINT `FK\_m2m\_class\_material\_class`

    FOREIGN KEY (`cla\_id`) REFERENCES `class` (`cla\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `m2m\_class\_material`

 ADD CONSTRAINT `FK\_m2m\_class\_material\_material`

    FOREIGN KEY (`mat\_id`) REFERENCES `material` (`mat\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `m2m\_user\_group`

 ADD CONSTRAINT `FK\_m2m\_user\_group\_group`

    FOREIGN KEY (`gro\_id`) REFERENCES `group` (`gro\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `m2m\_user\_group`

 ADD CONSTRAINT `FK\_m2m\_user\_group\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `message`

 ADD CONSTRAINT `FK\_message\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `message`

 ADD CONSTRAINT `FK\_message\_user\_02`

    FOREIGN KEY (`mes\_to\_user\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `news`

 ADD CONSTRAINT `FK\_news\_user`

    FOREIGN KEY (`use\_id`) REFERENCES `user` (`use\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `schedule`

 ADD CONSTRAINT `FK\_schedule\_class`

    FOREIGN KEY (`cla\_id`) REFERENCES `class` (`cla\_id`) ON DELETE Cascade ON UPDATE Cascade

;

ALTER TABLE `schedule`

 ADD CONSTRAINT `FK\_schedule\_place`

    FOREIGN KEY (`pla\_id`) REFERENCES `place` (`pla\_id`) ON DELETE Restrict ON UPDATE Cascade

;

ALTER TABLE `task`

 ADD CONSTRAINT `FK\_task\_class`

    FOREIGN KEY (`cla\_id`) REFERENCES `class` (`cla\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `user`

 ADD CONSTRAINT `FK\_user\_user\_type`

    FOREIGN KEY (`use\_type`) REFERENCES `user\_type` (`ust\_id`) ON DELETE Restrict ON UPDATE Restrict

;

SET FOREIGN\_KEY\_CHECKS=1

;

/\* Create Views \*/

CREATE OR REPLACE VIEW `not\_started\_courses` AS

SELECT

    course\_name.con\_name AS name,

    course\_level.col\_name AS language\_level,

    course.cou\_is\_individual AS is\_individual,

    course.cou\_date\_from AS date\_from,

    course.cou\_date\_to AS date\_to

FROM

    course

    INNER JOIN course\_level AS col ON con.col\_id = course.col\_id

    INNER JOIN course\_name AS con ON con.con\_id = course.con\_id

WHERE

    date\_from > NOW()

ORDER BY

    DATE\_FORMAT(date\_from, '%m%d') DESC;

;

CREATE OR REPLACE VIEW `top\_10\_tasks\_by\_comments\_count` AS

SELECT

    task.\*,

    COUNT(comment.com\_id) AS comments\_count

FROM

    task

    INNER JOIN comment ON comment.tas\_id = task.tas\_id

GROUP BY task.tas\_id

ORDER BY comments\_count

LIMIT 10;

;

CREATE OR REPLACE VIEW `top\_10\_tasks\_by\_marks` AS

SELECT

    task.\*,

    SUM(gradebook.gra\_mark) AS marks\_sum

FROM

    task

    INNER JOIN gradebook ON gradebook.tas\_id = task.tas\_id

GROUP BY task.tas\_id

ORDER BY marks\_sum

LIMIT 10;

;

CREATE OR REPLACE VIEW `top\_25\_news\_by\_date` AS

SELECT \* FROM news ORDER BY DATE\_FORMAT(new\_time, '%m%d%H%i%s');

;

CREATE OR REPLACE VIEW `top\_50\_users\_by\_marks` AS

SELECT

    user.\*,

    AVG(gradebook.gra\_mark) AS average\_mark

FROM

    user

    INNER JOIN gradebook ON gradebook.tas\_id = user.tas\_id

GROUP BY user.tas\_id

ORDER BY average\_mark

LIMIT 50;

;

CREATE OR REPLACE VIEW `users\_info` AS

SELECT

    user.\*,

    COUNT(message.use\_id) AS messages\_count,

    COUNT(answer.use\_id) AS answers\_count,

    COUNT(comment.use\_id) AS comments\_count,

    AVG(gradebook.gra\_mark) AS average\_mark

FROM

    user

    INNER JOIN message ON message.use\_id = user.use\_id

    INNER JOIN answer ON answer.use\_id = user.use\_id

    INNER JOIN comment ON comment.use\_id = user.use\_id

    INNER JOIN gradebook ON gradebook.use\_id = user.use\_id

GROUP BY user.use\_id

ORDER BY average\_mark;

;