Задание

Разработайте базу данных, автоматизирующую процесс работы кассира аэровокзала. Исходными данными для работы системы являются:

- номер авиарейса
- периодичность
- конечный пункт назначения
- дата и время отправления
- число мест

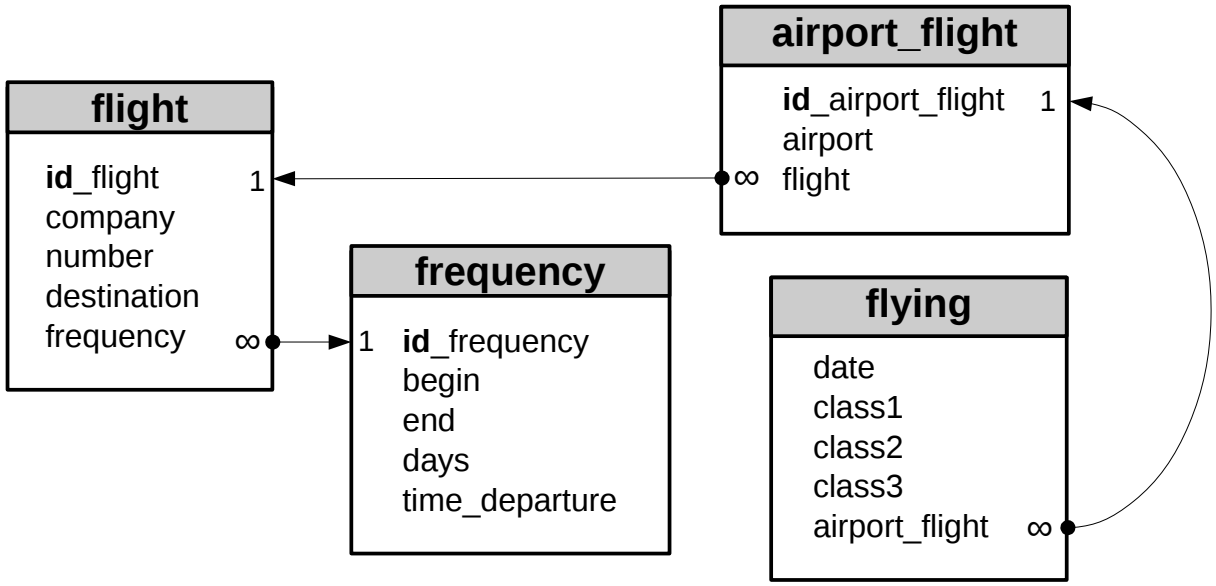
Кассир вводит в систему информацию от пассажира: дата, номер рейса, пункт назначения, количество мест. Если система находит места, отвечающие запросу, то она должна выдать информацию о номере места, количестве и типе мест. Если же запрос удовлетворить нельзя, то ищется альтернативный вариант. Система должна выдавать сводную справку по каждому аэропорту о наличии свободных мест на день, запрашиваемый пассажиром.

Построение концептуальной модели данных

Сущность	**Содержит**	**Описание сущности**
Рейс	код компании-перевозчика, номер рейса, конечный аэропорт (IATA-код)	осуществляется в строго заданные дни; не привязан к какому-либо конкретному аэропорту
Периодичность рейса	дата начала и конца действия рейса, дни недели (множество), время вылета	не привязана к какому-либо конкретному аэропорту
Полёты	дата, количество свободных мест по классам (упрощённо их 3)	все совершённые полёты с учётом рейса и начального аэропорта
Рейсы аэропорта	IATA-код аэропорта	содержит все рейсы любого аэропорта

Связи между сущностями:

полёты [∞] → [1] рейсы аэропорта [∞] → [1] рейсы [∞] → [1] периодичность



Перенос модели данных в среду целевой СУБД

Создание таблиц

```
CREATE SCHEMA if not exists airport;

CREATE TABLE `airport`.`flight` (
  `id_flight` INT NOT NULL AUTO_INCREMENT,
  `company` CHAR(2) NOT NULL,
  `number` SMALLINT(4) UNSIGNED NOT NULL,
  `destination` CHAR(3) NOT NULL,
  `frequency` INT NOT NULL,
  PRIMARY KEY (`id_flight`)
);

CREATE TABLE `airport`.`frequency` (
  `id_frequency` INT NOT NULL AUTO_INCREMENT,
  `begin` DATE NULL,
  `end` DATE NULL,
  `days` TINYINT NULL DEFAULT 127,
  `time_departure` TIME NULL,
  PRIMARY KEY (`id_frequency`)
);

CREATE TABLE `airport`.`airport_flight` (
  `id_airport_flight` INT NOT NULL AUTO_INCREMENT,
  `airport` CHAR(3) NOT NULL,
  `flight` INT NOT NULL,
  PRIMARY KEY (`id_airport_flight`)
);

CREATE TABLE `airport`.`flying` (
  `date` DATE NOT NULL,
  `class1` TINYINT UNSIGNED NULL,
  `class2` TINYINT UNSIGNED NULL,
  `class3` TINYINT UNSIGNED NULL,
  `airport_flight` INT NOT NULL
);
```

Создание связей между таблицами

```
ALTER TABLE `airport`.`flight`
  ADD INDEX `frequency_idx` (`frequency` ASC);
ALTER TABLE `airport`.`flight`
  ADD CONSTRAINT `frequency`  FOREIGN KEY (`frequency`)
    REFERENCES `airport`.`frequency` (`id_frequency`)
    ON DELETE NO ACTION  ON UPDATE NO ACTION;

ALTER TABLE `airport`.`airport_flight`
  ADD INDEX `flight_idx` (`flight` ASC);
ALTER TABLE `airport`.`airport_flight`
  ADD CONSTRAINT `flight`  FOREIGN KEY (`flight`)
    REFERENCES `airport`.`flight` (`id_flight`)
    ON DELETE CASCADE  ON UPDATE NO ACTION;

ALTER TABLE `airport`.`flying`
  ADD INDEX `airport_flight_idx` (`airport_flight` ASC);
ALTER TABLE `airport`.`flying`
  ADD CONSTRAINT `airport_flight`  FOREIGN KEY (`airport_flight`)
    REFERENCES `airport`.`airport_flight` (`id_airport_flight`)
    ON DELETE CASCADE  ON UPDATE NO ACTION;
```

Процедура для запроса свободных мест для использования при работе кассира

```
USE `airport`;
DROP procedure IF EXISTS `free_seats`;

DELIMITER $$
USE `airport`$$
CREATE PROCEDURE `free_seats`
  (base char(3), co char(2), num smallint unsigned, d date, count tinyint unsigned)
BEGIN
  if (base is null) then
    if (d is not null) then
      SELECT airport, class1, class2, class3 FROM flying, airport_flight, flight
        WHERE flight=id_flight and airport_flight=id_airport_flight
          and if(co, company=co, true) and if(num, number=num, true) and date=d
          and (class3>=count or class2>=count or class1>=count);
       else
      SELECT airport, class1, class2, class3, date FROM flying, airport_flight, flight
        WHERE flight=id_flight and airport_flight=id_airport_flight
          and if(co, company=co, true) and if(num, number=num, true)
          and (class3>=count or class2>=count or class1>=count);
    end if;
  else
    DROP TABLE if exists t;   # if last run failed
    CREATE TEMPORARY TABLE t
      SELECT airport, class1, class2, class3, date FROM flying, airport_flight, flight
        WHERE flight=id_flight and airport_flight=id_airport_flight
          and if(co, company=co, true) and if(num, number=num, true) and if(d, date=d, true)
          and (class3>=count or class2>=count or class1>=count);
    if (d is not null) then
      set @c1 = -1;
      SELECT class1, class2, class3 into @c1, @c2, @c3 FROM t WHERE airport=base;
      if (@c1 != -1)
        then SELECT @c1 as 'class1', @c2 as 'class2', @c3 as 'class3';
        else SELECT airport, class1, class2, class3 FROM t;  end if;
       else
      DROP TABLE if exists ft;   # if last run failed
      CREATE TEMPORARY TABLE ft
        SELECT * FROM t WHERE airport=base;
         SELECT count(*) into @ft_count FROM ft;
         if (@ft_count != 0)
        then SELECT class1, class2, class3, date FROM ft;
        else SELECT * FROM t;  end if;
      DROP TABLE ft;
    end if;
    DROP TABLE t;
  end if;
END $$
DELIMITER;
```

Тестирование процедуры запроса свободных мест

Результаты тестирования будут отображаться в следующем виде: сначала текст запроса, затем полученная в результате выполнения запроса таблица.

```
SELECT destination, airport, company, number, date, class1, class2, class3
FROM flying, airport_flight, flight
WHERE flight=id_flight and airport_flight=id_airport_flight
```

destination	airport	company	number	date	class1	class2	class3
KBP	DNK	PS	666	2015-05-22	10	3	1
KBP	DNK	PS	666	2015-05-25	13	12	10
KBP	ODS	PS	666	2015-05-22	7	1	0
KBP	ODS	PS	666	2015-05-25	14	18	17
LHR	KBP	BA	125	2015-05-21	0	0	0
LHR	KBP	BA	125	2015-05-22	15	17	3
LHR	KBP	BA	125	2015-05-23	13	10	18
LHR	ZHL	BA	125	2015-05-21	3	1	0

LHR	ZHL	BA	125	2015-05-22	7	9	14
LHR	ZHL	BA	125	2015-05-23	14	18	18

CALL free_seats(null, 'BA', '125', null, 1)

airport	class1	class2	class3	date
KBP	15	17	3	2015-05-22
KBP	13	10	18	2015-05-23
ZHL	3	1	0	2015-05-21
ZHL	7	9	14	2015-05-22
ZHL	14	18	18	2015-05-23

CALL free_seats(null, 'BA', '125', '2015-05-22', 1)

airport	class1	class2	class3
KBP	15	17	3
ZHL	7	9	14

CALL free_seats('KBP', null, null, '2015-05-23', 1)

class1	class2	class3
13	10	18

CALL free_seats('KBP', 'BA', '125', null, 1)

class1	class2	class3	date
15	17	3	2015-05-22
13	10	18	2015-05-23

CALL free_seats('KBP', 'BA', '125', '2015-05-22', 1)

class1	class2	class3
15	17	3