Скрипты создания таблиц и хранимых процедур БД InterBase 7.0.

Удаление таблиц

```
DROP TABLE TABLE_TEC;
DROP TABLE TABLE_FACT;
DROP TABLE TABLE_PR_MUS;
DROP TABLE TABLE_PR_NUJD;
DROP TABLE TABLE_FIO;
DROP TABLE TABLE_GOD;
```

Создание таблиц

```
CREATE TABLE "TABLE_FIO"
(
  "ID"      INTEGER NOT NULL,
  "KOD_PR"  INTEGER NOT NULL,
  "FIORUC"  CHAR(255) CHARACTER SET WIN1251 NOT NULL,
   CONSTRAINT "AK_300" UNIQUE ("KOD_PR"),
   CONSTRAINT "PK_101" PRIMARY KEY ("ID")
);
ALTER TABLE "TABLE_FIO" ADD CONSTRAINT "FK_201"
  FOREIGN KEY ("KOD_PR") REFERENCES TABLE_PR_NUJD ("ID");
COMMIT;
```

```
CREATE TABLE "TABLE_PR_NUJD"
(
  "ID"      INTEGER NOT NULL,
  "NAME_PR" CHAR(255) CHARACTER SET WIN1251 NOT NULL,
  "ADRES"   CHAR(255) CHARACTER SET WIN1251 NOT NULL,
  "RAION"   CHAR(255) CHARACTER SET WIN1251 NOT NULL,
   CONSTRAINT "PK_1" PRIMARY KEY ("ID")
);
```

```
CREATE TABLE "TABLE_PR_MUS"
(
  "ID"      INTEGER NOT NULL,
  "NAME_PR" CHAR(200) CHARACTER SET WIN1251 NOT NULL,
  "CENA_M3" FLOAT NOT NULL,
  CONSTRAINT "AK_302" UNIQUE ("NAME_PR"),
  CONSTRAINT "PK_104" PRIMARY KEY ("ID")
);
```

```
CREATE TABLE "TABLE_FACT"
(
  "ID"      INTEGER NOT NULL,
  "ID1"     INTEGER NOT NULL,
  "GOD"     INTEGER NOT NULL,
  "FLAG"    BOOLEAN DEFAULT false NOT NULL,
  "V" INTEGER NOT NULL,
   CONSTRAINT "PK_100" PRIMARY KEY ("ID")
);
ALTER TABLE "TABLE_FACT" ADD CONSTRAINT "FK_200"
  FOREIGN KEY ("ID1") REFERENCES TABLE_PR_NUJD ("ID");
COMMIT;
```

```
CREATE TABLE "TABLE_TEC"
(
  "ID"      INTEGER NOT NULL,
  "ID1"     INTEGER NOT NULL,
  "ID2"     INTEGER NOT NULL,
  "KOL_M3"  INTEGER NOT NULL,
  "CHISLO"  INTEGER NOT NULL,
  "MESAC"   INTEGER NOT NULL,
  "GOD"     INTEGER NOT NULL,
```

```
  "KOL_M3_ZAPL"   INTEGER NOT NULL,
   CONSTRAINT "PK_102" PRIMARY KEY ("ID")
);
ALTER TABLE "TABLE_TEC" ADD CONSTRAINT "FK_202" FOREIGN KEY ("ID1")
  REFERENCES TABLE_PR_MUS ("ID");
ALTER TABLE "TABLE_TEC" ADD CONSTRAINT "FK_203" FOREIGN KEY ("ID2")
  REFERENCES TABLE_PR_NUJD ("ID");
COMMIT;
```

```
CREATE TABLE "TABLE_GOD"
(
      "ID"  INTEGER NOT NULL UNIQUE,
      "GOD"  INTEGER NOT NULL UNIQUE
);
```

Процедура заполняет таблицы TABLE_FIO и TABLE_PR_NUJD

```
CREATE PROCEDURE PROC_PR_NUJD
(
   P_NAME_PR  CHAR(255),
   P_ADRES    CHAR(255),
   P_RAION    CHAR(255),
   P_FIO      CHAR(255)
)
AS
  DECLARE VARIABLE I INTEGER;
  DECLARE VARIABLE J INTEGER;
  DECLARE VARIABLE K INTEGER;
BEGIN
  IF (NOT EXISTS(SELECT * FROM TABLE_PR_NUJD))THEN
   BEGIN
     /***********ТАБЛИЦА TABLE_PR_NUJD - ПУСТА******************/
     I=1;
   END
  ELSE
   BEGIN
     /***********ТАБЛИЦА TABLE_PR_NUJD - НЕ ПУСТА***************/
     SELECT MAX(ID) FROM TABLE_PR_NUJD INTO :I;
     I=I+1;
   END
  /************ПРОВЕРЯЕМ НА УНИКАЛЬНОСТЬ ПРЕДПРИЯТИЕ************/
  K=0;
  IF (EXISTS(SELECT NAME_PR FROM TABLE_PR_NUJD WHERE NAME_PR=:P_NAME_PR)) THEN
    BEGIN
      K=K+1;
    END
  IF (EXISTS(SELECT RAION FROM TABLE_PR_NUJD WHERE RAION=:P_RAION)) THEN
    BEGIN
      K=K+1;
    END
  IF (EXISTS(SELECT ADRES FROM TABLE_PR_NUJD WHERE ADRES=:P_ADRES)) THEN
    BEGIN
      K=K+1;
    END
  /************ЕСЛИ K=3, ЗНАЧИТ ЭТО ОДНО И ТОЖЕ ПРЕДПРИЯТИЕ*******/
  IF (K<3) THEN
    BEGIN
     INSERT INTO TABLE_PR_NUJD (ID,NAME_PR,ADRES,RAION)
       VALUES (:I,:P_NAME_PR,:P_ADRES,:P_RAION);
     /************СВЕДЕНИЯ О РУКОВОДИТЕЛЕ************************/
     IF (NOT EXISTS(SELECT * FROM TABLE_FIO))THEN
      BEGIN
        /***********ТАБЛИЦА TABLE_FIO - ПУСТА**********************/
        J=1;
      END
     ELSE
```

```
       BEGIN
         SELECT MAX(ID) FROM TABLE_FIO INTO :J;
         J=J+1;
       END
     /*****ЕСЛИ У ПРЕДПРИЯТИЯ - 2 РУК., ТО ГЕНЕРИРУЕТСЯ ОШИБКА******/
     INSERT INTO TABLE_FIO (ID,KOD_PR,FIORUC) VALUES (:J,:I,:P_FIO);
    END
END
```

Пользователь добавляет текущие сведения

```
CREATE PROCEDURE PROC_TEC
(
   P_ID1         INTEGER,
   P_ID2         INTEGER,
   P_KOL_M3      INTEGER,
   P_CHISLO      INTEGER,
   P_MESAC       INTEGER,
   P_KOL_M3_ZAPL INTEGER
)
AS
 DECLARE VARIABLE I INTEGER;
 DECLARE VARIABLE GD INTEGER;
BEGIN
 SELECT GOD FROM TABLE_GOD
   WHERE ID=1 INTO GD;
 IF (NOT EXISTS(SELECT * FROM TABLE_TEC))
      THEN
       BEGIN
         I=1;
       END
 ELSE
   BEGIN
     SELECT MAX(ID) FROM TABLE_TEC INTO :I;
     I=I+1;
   END
/*****************************************************************/
INSERT INTO TABLE_TEC (ID,ID1,ID2,KOL_M3,CHISLO,MESAC,GOD,KOL_M3_ZAPL)
   VALUES (:I,:P_ID1,:P_ID2,:P_KOL_M3,:P_CHISLO,:P_MESAC,:GD,:P_KOL_M3_ZAPL);
END
```

Устанавливает текущий год

```
CREATE PROCEDURE PROC_GOD
(
   P_GOD         INTEGER
)
AS
BEGIN
  IF (NOT EXISTS(SELECT * FROM TABLE_GOD))
      THEN
       BEGIN
         INSERT INTO TABLE_GOD (ID,GOD)
           VALUES (1,:P_GOD);
       END
      ELSE
       BEGIN
         UPDATE TABLE_GOD
          SET GOD=:P_GOD WHERE ID=1;
       END
END
```

Добавляем новое предприятие, вывозящие мусор

```
CREATE PROCEDURE PROC_NEW_MUS
(
   P_NAME_PR     CHAR(255),
   P_CENA_M3     FLOAT
)
AS
  DECLARE VARIABLE I INTEGER;
BEGIN
  IF (NOT EXISTS(SELECT * FROM TABLE_PR_MUS))
      THEN
       BEGIN
         I=1;
       END
  ELSE
   BEGIN
     SELECT MAX(ID) FROM TABLE_PR_MUS INTO :I;
     I=I+1;
   END
  INSERT INTO TABLE_PR_MUS (ID,NAME_PR,CENA_M3)
   VALUES (:I,:P_NAME_PR,:P_CENA_M3);
END
```

Добавляем информацию и фактически вывозимом мусоре

```
CREATE PROCEDURE PROC_FACT
(
   P_ID1 INTEGER,
   P_GOD INTEGER,
   P_V   INTEGER
)
AS
  DECLARE VARIABLE I INTEGER;
BEGIN
  IF (NOT EXISTS(SELECT * FROM TABLE_FACT))
      THEN
       BEGIN
         I=1;
       END
  ELSE
   BEGIN
     SELECT MAX(ID) FROM TABLE_FACT INTO :I;
     I=I+1;
   END
  /*************************************************************/
  IF (NOT EXISTS(SELECT * FROM TABLE_FACT
                 WHERE (ID1=:P_ID1)AND(GOD=:P_GOD)))
      THEN
       BEGIN
        /***ПРОВЕРЯЕМ, ЧТОБЫ ID1 И GOD БЫЛИ РАЗНЫМИ У ДВУХ СТРОК****/
        INSERT INTO TABLE_FACT (ID,ID1,GOD,FLAG,V)
              VALUES (:I,:P_ID1,:P_GOD,FALSE,:P_V);
       END
END
```

Выполняем подсчет на конец года

```
CREATE PROCEDURE PROC_END_GOD
(
   P_END_GOD INTEGER
)
AS
  /********ДЛЯ ОРГАНИЗАЦИИ ЦИКЛА*******************************/
  DECLARE VARIABLE J1 INTEGER;
  DECLARE VARIABLE I1 INTEGER;
  DECLARE VARIABLE I INTEGER;
  DECLARE VARIABLE J INTEGER;
  DECLARE VARIABLE GODS INTEGER; /*ТЕКУЩИЙ ГОД*/
  DECLARE VARIABLE IDS1 INTEGER; /*ТЕКУЩИЙ КОД ПРЕДПРИЯТИЯ*/
  DECLARE VARIABLE FL BOOLEAN;
  DECLARE VARIABLE VI INTEGER;
  DECLARE VARIABLE P INTEGER;
BEGIN
 J=1;
 SELECT COUNT(*) FROM TABLE_FACT INTO :I; /*I=ЧИСЛО ЗАПИСЕЙ*/
 WHILE (J<=I) DO
  BEGIN
   /**************РАБОТАЕМ С ОДНОЙ ЗАПИСЬЮ**********************/
   SELECT FLAG FROM TABLE_FACT WHERE ID=:J INTO :FL;
   SELECT GOD FROM TABLE_FACT WHERE ID=:J INTO :GODS;
   SELECT ID1 FROM TABLE_FACT WHERE ID=:J INTO :IDS1;
   IF (FL=FALSE) THEN
     BEGIN
       UPDATE TABLE_FACT
        SET FLAG=TRUE WHERE ID=:J;
       /***************СЧИТАЕМ ОБЪЕМ*****************************/
       VI=0;
       J1=1;
       SELECT COUNT(*) FROM TABLE_TEC INTO :I1;
       WHILE (J1<=I1) DO
         BEGIN
           IF (EXISTS(SELECT KOL_M3 FROM TABLE_TEC
                      WHERE (ID=:J1)AND(ID2=:IDS1)AND(GOD=:GODS))) THEN
             BEGIN
                SELECT KOL_M3 FROM TABLE_TEC
                  WHERE (ID=:J1)AND(ID2=:IDS1)AND(GOD=:GODS)
                    INTO :P;
                VI=VI+P;
             END
           J1=J1+1;
         END
       /*********************************************************/
       UPDATE TABLE_FACT
          SET V=:VI WHERE ID=:J;
     END
   /*************************************************************/
   J=J+1;
  END
END
```

Возвращаем значение текущего года

```
CREATE PROCEDURE PROC_RETURN_GOD
RETURNS (P_GOD INTEGER)
AS
BEGIN
  SELECT GOD FROM TABLE_GOD WHERE ID=1 INTO :P_GOD;
  SUSPEND;
END
```

Возвращаем задолжность по оплате

```
CREATE PROCEDURE PROC_RETURN_ZA1
(
   P_GOD INTEGER,
   P_KOD INTEGER
)
RETURNS (SUMI FLOAT)
AS
  DECLARE VARIABLE I    INTEGER;
  DECLARE VARIABLE J    INTEGER;
  DECLARE VARIABLE IDS1 INTEGER;
  DECLARE VARIABLE H1   INTEGER;
  DECLARE VARIABLE H2   INTEGER;
  DECLARE VARIABLE CEN   FLOAT;
BEGIN
 J=1;
 SUMI=0;
 SELECT COUNT(*) FROM TABLE_TEC INTO :I; /*I=ЧИСЛО ЗАПИСЕЙ*/
 WHILE (J<=I) DO
  BEGIN
    IF (EXISTS(SELECT * FROM TABLE_TEC
                WHERE (GOD=:P_GOD)AND(ID2=:P_KOD)AND(ID=:J))) THEN
      BEGIN
        SELECT ID1 FROM TABLE_TEC
          WHERE (GOD=:P_GOD)AND(ID2=:P_KOD)AND(ID=:J) INTO :IDS1;
        SELECT CENA_M3 FROM TABLE_PR_MUS
          WHERE ID=:IDS1 INTO :CEN;
        SELECT KOL_M3 FROM TABLE_TEC
          WHERE (GOD=:P_GOD)AND(ID2=:P_KOD)AND(ID=:J) INTO :H1;
        SELECT KOL_M3_ZAPL FROM TABLE_TEC
          WHERE (GOD=:P_GOD)AND(ID2=:P_KOD)AND(ID=:J) INTO :H2;
        IF (H1-H2>0) THEN
          BEGIN
            SUMI=SUMI+(H1-H2)*CEN;
          END
      END
    J=J+1;
  END
  SUSPEND;
END
```

Удаление процедур

```
DROP PROCEDURE PROC_GOD;
DROP PROCEDURE PROC_NEW_MUS;
DROP PROCEDURE PROC_TEC;
DROP PROCEDURE PROC_FACT
DROP PROCEDURE PROC_END_GOD;
DROP PROCEDURE PROC_RETURN_GOD;
DROP PROCEDURE PROC_PR_NUJD;
DROP PROCEDURE PROC_RETURN_ZA1
```

Спроектированная база данных