

Queries Used to Read NIBRS Data into ACCESS 2000

The Justice Research and Statistics Association Incident-Based Resource Center
www.jrsa.org/ibrrc

Queries are used to read the FBI NIBRS data into ACCESS 2000. These queries can be downloaded as part of the [ACCESS 2000 Executable File](#). The following code creates the tables and variables necessary to read NIBRS data into ACCESS. If you need any assistance working with the information provided, [please contact us](#) at ibrrc@jrsa.org.

ADMINISTRATIVE Segment

Query **mkADMINISTRATIVE_Table** is a data definition query that creates the ADMINISTRATIVE table structure. The table fields are named and data types are set when this query is executed.

```
CREATE TABLE ADMINISTRATIVE (SEG TEXT(2), UID TEXT(21), INC_DT DATE, RPTDT_INDIC TEXT(1), INC_HR TEXT(2), TOT_OFF_SEG INTEGER, TOT_VIC_SEG INTEGER, TOT_OFN_SEG INTEGER, TOT_ARR_SEG INTEGER, CITY_SUB TEXT(4), CLR_EX TEXT(1), CLR_EX_DT DATE)
```

Query **mkADMINISTRATIVE_Index** creates the ADMINISTRATIVE table index. Field UID is used as the index. This same field is also set as the key field for the table.

```
CREATE INDEX PrimaryKey ON ADMINISTRATIVE(UID) WITH PRIMARY
```

Query **appADMINISTRATIVE_Data** is an append query that uses the NIBRS file record layout to format the data from segment-level file SEG01YY.txt and load it into table ADMINISTRATIVE (SEG01YY.txt is known to the query as SEG01).

```
INSERT INTO ADMINISTRATIVE ( SEG, UID, INC_DT, RPTDT_INDIC, INC_HR, TOT_OFF_SEG, TOT_VIC_SEG, TOT_OFN_SEG, TOT_ARR_SEG, CITY_SUB, CLR_EX, CLR_EX_DT )
SELECT IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)), IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(MID([Field1],26,8)=' ',Null,DATEVALUE(MID([Field1],30,2) & '/' & MID([Field1],32,2) & '/' & MID([Field1],26,4))),
IIF(MID([Field1],34,1)=' ',Null,MID([Field1],34,1)), IIF(MID([Field1],35,2)=' ',Null,MID([Field1],35,2)),
IIF(MID([Field1],37,2)=' ',Null,VAL(MID([Field1],37,2))), IIF(MID([Field1],39,3)=' ',Null,VAL(MID([Field1],39,3))),
IIF(MID([Field1],42,2)=' ',Null,VAL(MID([Field1],42,2))), IIF(MID([Field1],44,2)=' ',Null,VAL(MID([Field1],44,2))),
IIF(MID([Field1],46,4)=' ',Null,MID([Field1],46,4)), IIF(MID([Field1],50,1)=' ',Null,MID([Field1],50,1)),
IIF(MID([Field1],51,8)=' ',Null,DATEVALUE(MID([Field1],55,2) & '/' & MID([Field1],57,2) & '/' & MID([Field1],51,4)))
FROM SEG01;
```

OFFENSE Segment

Query **mkOFFENSE_Table** creates the OFFENSE table structure. The table fields are named and data types are set when this query is executed. The relationship with the ADMINISTRATIVE table is also created by linking the key field UID in table OFFENSE with the key field UID in the ADMINISTRATIVE table.

```
CREATE TABLE OFFENSE (SEG TEXT(2), UID TEXT(21), OFF_CODE TEXT(3), ATT_COMP TEXT(1), USING1 TEXT(1), USING2 TEXT(2), USING3 TEXT(3), LOC TEXT(2), NO_PREM_ENT LONG, METH_ENT TEXT(1), CRIM_ACT1 TEXT(1), CRIM_ACT2 TEXT(1), CRIM_ACT3 TEXT(1), WEPF1 TEXT(2), AUTO1 TEXT(1), WEPF2 TEXT(2), AUTO2 TEXT(1), WEPF3 TEXT(2), AUTO3 TEXT(1), BIAS TEXT(2), CONSTRAINT fkOFFENSEADMIN FOREIGN KEY (UID) REFERENCES ADMINISTRATIVE)
```

Query **mkOFFENSE_Index** creates the OFFENSE table index. Fields UID and OFF_CODE are set as both the index and the key fields for table OFFENSE.

```
CREATE INDEX PrimaryKey ON OFFENSE(UID, OFF_CODE) WITH PRIMARY
```

Query **appOFFENSE_Data** uses the NIBRS file record layout to format the data from segment-level file SEG02YY.txt and load it into table OFFENSE (SEG02YY.txt is known to the query as SEG02).

```
INSERT INTO OFFENSE ( SEG, UID, OFF_CODE, ATT_COMP, USING1, USING2, USING3, LOC, NO_PREM_ENT, METH_ENT, CRIM_ACT1, CRIM_ACT2, CRIM_ACT3, WEPF1, AUTO1, WEPF2, AUTO2, WEPF3, AUTO3, BIAS )
SELECT IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)), IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(MID([Field1],34,3)=' ',Null,MID([Field1],34,3)), IIF(MID([Field1],37,1)=' ',Null,MID([Field1],37,1)),
IIF(Mid([Field1],38,1)=' ',Null,MID([Field1],38,1)), IIF(Mid([Field1],39,1)=' ',Null,MID([Field1],39,1)),
IIF(Mid([Field1],40,1)=' ',Null,MID([Field1],40,1)), IIF(Mid([Field1],41,2)=' ',Null,MID([Field1],41,2)),
IIF(MID([Field1],43,2)=' ',Null,MID([Field1],43,2)), IIF(MID([Field1],45,1)=' ',Null,MID([Field1],45,1)),
IIF(Mid([Field1],46,1)=' ',Null,MID([Field1],46,1)), IIF(Mid([Field1],47,1)=' ',Null,MID([Field1],47,1)),
IIF(Mid([Field1],48,1)=' ',Null,MID([Field1],48,1)), IIF(Mid([Field1],49,2)=' ',Null,MID([Field1],49,2)),
IIF(Mid([Field1],51,1)=' ',Null,MID([Field1],51,1)), IIF(Mid([Field1],52,2)=' ',Null,MID([Field1],52,2)),
IIF(Mid([Field1],54,1)=' ',Null,MID([Field1],54,1)), IIF(Mid([Field1],55,2)=' ',Null,MID([Field1],55,2)),
IIF(Mid([Field1],57,1)=' ',Null,MID([Field1],57,1)), IIF(MID([Field1],58,2)=' ',Null,mid([Field1],58,2))
FROM SEG02
```

Query **mkOFFENSE_ACTIVITY_Table** creates the OFFENSE_ACTIVITY table structure. The table fields are named and data types are set when this query is executed. The relationship with table OFFENSE is created by linking key fields UID and OFF_CODE in table OFFENSE_ACTIVITY with the same key fields in table OFFENSE.

```
CREATE TABLE OFFENSE_ACTIVITY(UID TEXT(21), OFF_CODE TEXT(3), ACT_SEQ_NO LONG, CRIM_ACT TEXT(1), CONSTRAINT fkOFFENSEACTIVITY FOREIGN KEY (UID, OFF_CODE) REFERENCES OFFENSE)
```

Query **mkOFFENSE_ACTIVITY Index** creates the OFFENSE_ACTIVITY table index. Fields UID and ACT_SEQ_NO are used as the index for the table OFFENSE_ACTIVITY.

```
CREATE INDEX OFFENSE_ACTIVITY ON OFFENSE_ACTIVITY(UID, ACT_SEQ_NO)
```

Queries **appOFFENSE_ACTIVITY_Data01** through **appOFFENSE_ACTIVITY_Data03** copy the data from fields UID, CRIM_ACT1, CRIM_ACT2 and CRIM_ACT3 in table OFFENSE and loads the data into table OFFENSE_ACTIVITY field CRIM_ACT. The criminal activity sequence number, ACT_SEQ_NO, is loaded by this query as well.

appOFFENSE_ACTIVITY_Data01:

```
INSERT INTO OFFENSE_Activity ( UID, OFF_CODE, ACT_SEQ_NO, CRIM_ACT )
SELECT [OFFENSE].[UID], [OFF_CODE], 1, [OFFENSE].CRIM_ACT1
FROM OFFENSE
WHERE [OFFENSE].CRIM_ACT1 Is Not Null
```

appOFFENSE_ACTIVITY_Data02

```
INSERT INTO OFFENSE_Activity ( UID, OFF_CODE, ACT_SEQ_NO, CRIM_ACT )
SELECT [OFFENSE].[UID], [OFF_CODE], 2, [OFFENSE].CRIM_ACT2
FROM OFFENSE
WHERE [OFFENSE].CRIM_ACT2 Is Not Null
```

appOFFENSE_ACTIVITY_Data03

```
INSERT INTO OFFENSE_Activity ( UID, OFF_CODE, ACT_SEQ_NO, CRIM_ACT )
SELECT [OFFENSE].[UID], [OFF_CODE], 3, [OFFENSE].CRIM_ACT3
FROM OFFENSE
WHERE [OFFENSE].CRIM_ACT3 Is Not Null
```

Query **mkOFFENSE_USING_Table** creates the OFFENSE_USING table structure. The table fields are named and data types are set when this query is executed. The relationship with table OFFENSE is created by linking key fields UID and OFF_CODE in table OFFENSE_USING with the same key fields in table OFFENSE.

```
CREATE TABLE OFFENSE_USING(UID TEXT(21), OFF_CODE TEXT(3), USING_SEQ_NO LONG, USING TEXT(1),
CONSTRAINT fkOFFENSEUSING FOREIGN KEY (UID, OFF_CODE) REFERENCES OFFENSE)
```

Query **mkOFFENSE_USING_Index** creates the OFFENSE_USING table index. Fields UID, OFF_CODE and USING_SEQ_NO are used as the index fields for table OFFENSE_USING.

```
CREATE INDEX OFFENSE_USING ON OFFENSE_USING(UID, OFF_CODE, USING_SEQ_NO)
```

Queries **appOFFENSE_USING_Data01** through **appOFFENSE_USING_Data03** copy the data from fields UID, USING1, USING2 and USING3 in table OFFENSE and loads them into table OFFENSE_USING field USING. The using sequence number USING_SEQ_NO, is also loaded by this query.

appOFFENSE_USING_Data01:

```
INSERT INTO OFFENSE_USING ( UID, OFF_CODE, USING_SEQ_NO, [USING] )
SELECT [OFFENSE].[UID], [OFFENSE].[OFF_CODE], 1, [OFFENSE].USING1
FROM OFFENSE
WHERE USING1 IS NOT NULL;
```

appOFFENSE_USING_Data02:

```
INSERT INTO OFFENSE_USING ( UID, OFF_CODE, USING_SEQ_NO, [USING] )
SELECT [OFFENSE].[UID], [OFFENSE].[OFF_CODE], 2, [OFFENSE].USING2
FROM OFFENSE
WHERE USING2 IS NOT NULL;
```

appOFFENSE_USING_Data03:

```
INSERT INTO OFFENSE_USING ( UID, OFF_CODE, USING_SEQ_NO, [USING] )
SELECT [OFFENSE].[UID], [OFFENSE].[OFF_CODE], 3, [OFFENSE].USING3
FROM OFFENSE
WHERE USING3 IS NOT NULL;
```

Query **mkOFFENSE_WEAPON_Table** creates the OFFENSE_WEAPON table structure. The table fields are named and data types are set when this query is executed. The relationship with table OFFENSE is created by linking key fields UID and OFF_CODE in table OFFENSE_WEAPON with the same key fields in table OFFENSE.

```
CREATE TABLE OFFENSE_WEAPON (UID TEXT(21), OFF_CODE TEXT(3), WEPF_SEQ_NO LONG, WEPF
TEXT(2), AUTO TEXT(1), CONSTRAINT fkOFFENSE_WEAPON FOREIGN KEY (UID, OFF_CODE) REFERENCES
OFFENSE)
```

Query **mkOFFENSE_WEAPON_Index** creates the OFFENSE_WEAPON table index. Fields UID and WEPF_SEQ_NO are set as the index fields for table OFFENSE_USING.

```
CREATE INDEX OFFENSE_WEAPON on OFFENSE_WEAPON(UID, WEPF_SEQ_NO)
```

Queries **appOFFENSE_WEAPON_Data01** through **appOFFENSE_WEAPON_Data03** copy the data from fields UID, OFF_CODE, WEPF1, AUTO1, WEPF2, AUTO2, WEPF3, and AUTO3 in table OFFENSE and loads the data into table OFFENSE_WEAPON. The weapon/force sequence number, WEPF_SEQ_NO, is also loaded by this query.

appOFFENSE_WEAPON_Data01:

```
INSERT INTO OFFENSE_WEAPON ( UID, OFF_CODE, WEPF_SEQ_NO, WEPF, AUTO )
SELECT [OFFENSE].[UID], [OFF_CODE], 1, [OFFENSE].WEPF1, [OFFENSE].AUTO1
FROM OFFENSE
WHERE WEPF1 IS NOT NULL;
```

appOFFENSE_WEAPON_Data02:

```
INSERT INTO OFFENSE_WEAPON ( UID, OFF_CODE, WEPF_SEQ_NO, WEPF, AUTO )
SELECT [OFFENSE].[UID], [OFF_CODE], 2, [OFFENSE].WEPF2, [OFFENSE].AUTO2
FROM OFFENSE
WHERE WEPF2 IS NOT NULL;
```

appOFFENSE_WEAPON_Data03:

```
INSERT INTO OFFENSE_WEAPON ( UID, OFF_CODE, WEPF_SEQ_NO, WEPF, AUTO )
SELECT [OFFENSE].UID, [OFF_CODE], 3, [OFFENSE].WEPF3, [OFFENSE].AUTO3
FROM OFFENSE
WHERE WEPF3 IS NOT NULL;
```

Query **mkDrop_Normalized_Fields_From_OFFENSE** deletes from table OFFENSE the fields of data now stored in tables OFFENSE_ACTIVITY, OFFENSE_USING and OFFENSE_WEAPON.

```
ALTER TABLE OFFENSE DROP COLUMN CRIM_ACT1, CRIM_ACT2, CRIM_ACT3, USING1, USING2, USING3,
WEPF1, AUTO1, WEPF2, AUTO2, WEPF3, AUTO3
```

PROPERTY Segment

Query **mkPROPERTY_Table** creates the PROPERTY table structure. The table fields are named and data types are set when this query is executed. The relationship with table ADMINISTRATIVE is also created by linking key field UID in table PROPERTY with key field UID in table ADMINISTRATIVE.

```
CREATE TABLE PROPERTY (SEG TEXT(2), UID TEXT(21), TYPE_LOSS TEXT(1), DESCR TEXT(2), VALU
CURRENCY, DATE_RECOV DATE, NO_STOL_MV LONG, NO_RECOV_MV LONG, DRUG1 TEXT(1),
WHOLE_QUANT1 LONG, FRAC_QUANT1 DOUBLE, MEAS1 TEXT(2), DRUG2 TEXT(1), WHOLE_QUANT2 LONG,
FRAC_QUANT2 DOUBLE, MEAS2 TEXT(2), DRUG3 TEXT(1), WHOLE_QUANT3 LONG, FRAC_QUANT3 DOUBLE,
MEAS3 TEXT(2), CONSTRAINT fkPROPERTYADMIN FOREIGN KEY (UID) REFERENCES ADMINISTRATIVE)
```

Query **mkPROPERTY_Index** creates the PROPERTY table index. PROPERTY table fields UID, TYPE_LOSS and DESCR are used as the index fields for table PROPERTY. These same fields also become the key fields for table PROPERTY once this query is executed.

```
CREATE INDEX PrimaryKey ON PROPERTY(UID, TYPE_LOSS, DESCR) WITH PRIMARY
```

Query **appPROPERTY_Data** uses the NIBRS file record layout to format the data from segment-level file SEG03YY.txt and load it into table PROPERTY (SEG03YY.txt is known to the query as SEG03):

```
INSERT INTO PROPERTY ( SEG, UID, TYPE_LOSS, DESCR, VALU, DATE_RECOV, NO_STOL_MV,
NO_RECOV_MV, DRUG1, WHOLE_QUANT1, FRAC_QUANT1, MEAS1, DRUG2, WHOLE_QUANT2,
FRAC_QUANT2, MEAS2, DRUG3, WHOLE_QUANT3, FRAC_QUANT3, MEAS3 )
SELECT
IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)), IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(MID([Field1],34,1)=' ',Null,MID([Field1],34,1)), IIF(MID([Field1],35,2)=' ',Null,MID([Field1],35,2)),
IIF(Mid([Field1],37,9)=' ',Null,VAL(MID([Field1],37,9))),
IIF(Mid([Field1],46,8)=' ',Null,DATEVALUE(MID([Field1],50,2) & '/' & MID([Field1],52,2) & '/' & MID([Field1],46,4))),
IIF(MID([Field1],54,2)=' ',Null,MID([Field1],54,2)),
IIF(MID([Field1],56,2)=' ',Null,MID([Field1],56,2)), IIF(Mid([Field1],58,1)=' ',Null,MID([Field1],58,1)),
IIF(MID([Field1],59,9)=' ',Null,VAL(MID([Field1],59,9))), IIF(Mid([Field1],68,3)=' ',Null,VAL(MID([Field1],68,3)/1000)),
IIF(Mid([Field1],71,2)=' ',Null,MID([Field1],71,2)), IIF(Mid([Field1],73,1)=' ',Null,MID([Field1],73,1)),
IIF(MID([Field1],74,9)=' ',Null,VAL(MID([Field1],74,9))),
IIF(Mid([Field1],83,3)=' ',Null,VAL(MID([Field1],83,3)/1000)),
IIF(Mid([Field1],86,2)=' ',Null,MID([Field1],86,2)), IIF(Mid([Field1],88,1)=' ',Null,MID([Field1],88,1)),
IIF(MID([Field1],89,9)=' ',Null,VAL(MID([Field1],89,9))),
IIF(Mid([Field1],98,3)=' ',Null,VAL(MID([Field1],98,3)/1000)), IIF(Mid([Field1],101,2)=' ',Null,MID([Field1],101,2))
FROM SEG03;
```

Query **mkPROPERTY_DRUG_Table** creates the PROPERTY_DRUG table structure. The table fields are named and data types are set when this query is executed. The relationship with table PROPERTY is created by linking key fields UID, TYPE_LOSS and DESCR in table PROPERTY_WEAPON with the same key fields in table PROPERTY.

```
CREATE TABLE PROPERTY_DRUG(UID TEXT(21), TYPE_LOSS TEXT(1), DESCR TEXT(2), DRUG_SEQ_NO
LONG, DRUG TEXT(1), DRUG_QUANT DOUBLE, MEAS TEXT(2), CONSTRAINT fkPROPERTYDRUG FOREIGN
KEY(UID, TYPE_LOSS, DESCR) REFERENCES PROPERTY)
```

Query **mkPROPERTY_DRUG_Index** creates the PROPERTY_DRUG table index. Fields UID and DRUG_SEQ_NO are used as the index fields for table PROPERTY_DRUG.

```
CREATE INDEX PROPERTY_DRUG ON PROPERTY_DRUG(UID, DRUG_SEQ_NO)
```

Queries **appPROPERTY_DRUG_Data01** through **appPROPERTY_DRUG_Data03** copy the data from fields UID, TYPE_LOSS, DESCR, DRUG1, DRUG2, DRUG3, WHOLE_QUANT1, WHOLE_QUANT2, WHOLE_QUANT3, FRAC_QUANT1, FRAC_QUANT2, FRAC_QUANT3, MEAS1, MEAS2, MEAS3 in table OFFENSE and loads the data into table PROPERTY_DRUG. The drug sequence number, DRUG_SEQ_NO, is also loaded by these queries.

appPROPERTY_DRUG_Data01:

```
INSERT INTO PROPERTY_DRUG ( UID, TYPE_LOSS, DESCR, DRUG_SEQ_NO, DRUG, DRUG_QUANT, MEAS )
SELECT [PROPERTY].UID, [PROPERTY].TYPE_LOSS, [PROPERTY].DESCR, 1, [PROPERTY].DRUG1,
([PROPERTY].WHOLE_QUANT1+[PROPERTY].FRAC_QUANT1), [PROPERTY].MEAS1
FROM PROPERTY
WHERE DRUG1 IS NOT NULL;
```

appPROPERTY_DRUG_Data02:

```
INSERT INTO PROPERTY_DRUG ( UID, TYPE_LOSS, DESCR, DRUG_SEQ_NO, DRUG, DRUG_QUANT, MEAS )
SELECT [PROPERTY].UID, [PROPERTY].TYPE_LOSS, [PROPERTY].DESCR, 2, [PROPERTY].DRUG2,
([PROPERTY].WHOLE_QUANT2+[PROPERTY].FRAC_QUANT2), [PROPERTY].MEAS2
FROM PROPERTY
WHERE DRUG2 IS NOT NULL;
```

appPROPERTY_DRUG_Data03:

```
INSERT INTO PROPERTY_DRUG ( UID, TYPE_LOSS, DESCR, DRUG_SEQ_NO, DRUG, DRUG_QUANT, MEAS )
SELECT [PROPERTY].UID, [PROPERTY].TYPE_LOSS, [PROPERTY].DESCR, 3, [PROPERTY].DRUG3,
([PROPERTY].WHOLE_QUANT3+[PROPERTY].FRAC_QUANT3), [PROPERTY].MEAS3
FROM PROPERTY
WHERE DRUG3 IS NOT NULL;
```

Query **mkDrop_Normalized_Fields_From_PROPERTY** deletes from table PROPERTY the fields of data now stored in table PROPERTY_DRUG.

```
ALTER TABLE PROPERTY DROP COLUMN DRUG1, WHOLE_QUANT1, FRAC_QUANT1, MEAS1, DRUG2,
WHOLE_QUANT2, FRAC_QUANT2, MEAS2, DRUG3, WHOLE_QUANT3, FRAC_QUANT3, MEAS3
```

VICTIM Segment

Query **mkVICTIM_Table** creates the VICTIM table structure. The table fields are named and data types are set when this query is executed. The relationship with table ADMINISTRATIVE is also created by linking key field UID in table VICTIM with key field UID in table ADMINISTRATIVE.

```
CREATE TABLE VICTIM (SEG TEXT(2), UID TEXT(21), VIC_SEQ_NO LONG, VICOFF1 TEXT(3), VICOFF2
TEXT(3), VICOFF3 TEXT(3), VICOFF4 TEXT(3), VICOFF5 TEXT(3), VICOFF6 TEXT(3), VICOFF7 TEXT(3),
VICOFF8 TEXT(3), VICOFF9 TEXT(3), VICOFF10 TEXT(3), VIC_TYPE TEXT(1), VIC_AGE TEXT(2), VIC_SEX
TEXT(1), VIC_RACE TEXT(1), VIC_ETHN TEXT(1), VIC_RESID TEXT(1), CIRCUM1 TEXT(2), CIRCUM2 TEXT(2),
ADD_CIRCUM TEXT(1), INJ1 TEXT(1), INJ2 TEXT(1), INJ3 TEXT(1), INJ4 TEXT(1), INJ5 TEXT(1), OFN1
LONG, VOR1 TEXT(2), OFN2 LONG, VOR2 TEXT(2), OFN3 LONG, VOR3 TEXT(2), OFN4 LONG, VOR4
TEXT(2), OFN5 LONG, VOR5 TEXT(2), OFN6 LONG, VOR6 TEXT(2), OFN7 LONG, VOR7 TEXT(2), OFN8 LONG,
VOR8 TEXT(2), OFN9 LONG, VOR9 TEXT(2), OFN10 LONG, VOR10 TEXT(2), CONSTRAINT fkVICTIMADMIN
FOREIGN KEY (UID) REFERENCES ADMINISTRATIVE)
```

Query **mkVICTIM_Index** creates the VICTIM table index. VICTIM table fields UID, and VIC_SEQ_NO are used as the index fields for table VICTIM. These same fields also become the key fields for table VICTIM once this query is executed.

```
CREATE INDEX PrimaryKey ON VICTIM(UID, VIC_SEQ_NO) WITH PRIMARY
```

Query **appVICTIM_Data** uses the NIBRS file record layout to format the data from segment-level file SEG04YY.txt and load it into table PROPERTY (SEG04YY.txt is known to the query as SEG04).

```
INSERT INTO VICTIM ( SEG, UID, VIC_SEQ_NO, VICOFF1, VICOFF2, VICOFF3, VICOFF4, VICOFF5, VICOFF6,
VICOFF7, VICOFF8, VICOFF9, VICOFF10, VIC_TYPE, VIC_AGE, VIC_SEX, VIC_RACE, VIC_ETHN, VIC_RESID,
CIRCUM1, CIRCUM2, ADD_CIRCUM, INJ1, INJ2, INJ3, INJ4, INJ5, OFN1, VOR1, OFN2, VOR2, OFN3, VOR3,
OFN4, VOR4, OFN5, VOR5, OFN6, VOR6, OFN7, VOR7, OFN8, VOR8, OFN9, VOR9, OFN10, VOR10 )
SELECT IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)),
IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)), IIF(MID([Field1],34,3)=' ',Null,VAL(MID([Field1],34,3))),
IIF(MID([Field1],37,3)=' ',Null,MID([Field1],37,3)), IIF(MID([Field1],40,3)=' ',Null,MID([Field1],40,3)),
IIF(MID([Field1],43,3)=' ',Null,MID([Field1],43,3)), IIF(MID([Field1],46,3)=' ',Null,MID([Field1],46,3)),
IIF(MID([Field1],49,3)=' ',Null,MID([Field1],49,3)), IIF(MID([Field1],52,3)=' ',Null,MID([Field1],52,3)),
IIF(MID([Field1],55,3)=' ',Null,MID([Field1],55,3)), IIF(MID([Field1],58,3)=' ',Null,MID([Field1],58,3)),
IIF(MID([Field1],61,3)=' ',Null,MID([Field1],61,3)), IIF(MID([Field1],64,3)=' ',Null,MID([Field1],64,3)),
IIF(MID([Field1],67,1)=' ',Null,MID([Field1],67,1)), IIF(MID([Field1],68,2)=' ',Null,MID([Field1],68,2)),
IIF(MID([Field1],70,1)=' ',Null,MID([Field1],70,1)), IIF(MID([Field1],71,1)=' ',Null,MID([Field1],71,1)),
IIF(MID([Field1],72,1)=' ',Null,MID([Field1],72,1)), IIF(MID([Field1],73,1)=' ',Null,MID([Field1],73,1)),
IIF(MID([Field1],74,2)=' ',Null,MID([Field1],74,2)), IIF(MID([Field1],76,2)=' ',Null,MID([Field1],76,2)),
IIF(MID([Field1],78,1)=' ',Null,MID([Field1],78,1)), IIF(MID([Field1],79,1)=' ',Null,MID([Field1],79,1)),
IIF(MID([Field1],80,1)=' ',Null,MID([Field1],80,1)), IIF(MID([Field1],81,1)=' ',Null,MID([Field1],81,1)),
IIF(MID([Field1],82,1)=' ',Null,MID([Field1],82,1)), IIF(MID([Field1],83,1)=' ',Null,MID([Field1],83,1)),
IIF(MID([Field1],84,2)=' ',Null,VAL(MID([Field1],84,2))), IIF(MID([Field1],86,2)=' ',Null,MID([Field1],86,2)),
IIF(MID([Field1],88,2)=' ',Null,VAL(MID([Field1],88,2))), IIF(MID([Field1],90,2)=' ',Null,MID([Field1],90,2)),
IIF(MID([Field1],92,2)=' ',Null,VAL(MID([Field1],92,2))), IIF(MID([Field1],94,2)=' ',Null,MID([Field1],94,2)),
IIF(MID([Field1],96,2)=' ',Null,VAL(MID([Field1],96,2))), IIF(MID([Field1],98,2)=' ',Null,MID([Field1],98,2)),
IIF(MID([Field1],100,2)=' ',Null,VAL(MID([Field1],100,2))), IIF(MID([Field1],102,2)=' ',Null,MID([Field1],102,2)),
IIF(MID([Field1],104,2)=' ',Null,VAL(MID([Field1],104,2))), IIF(MID([Field1],106,2)=' ',Null,MID([Field1],106,2)),
IIF(MID([Field1],108,2)=' ',Null,VAL(MID([Field1],108,2))), IIF(MID([Field1],110,2)=' ',Null,MID([Field1],110,2)),
IIF(MID([Field1],112,2)=' ',Null,VAL(MID([Field1],112,2))), IIF(MID([Field1],114,2)=' ',Null,MID([Field1],114,2)),
IIF(MID([Field1],116,2)=' ',Null,VAL(MID([Field1],116,2))), IIF(MID([Field1],118,2)=' ',Null,MID([Field1],118,2)),
IIF(MID([Field1],120,2)=' ',Null,VAL(MID([Field1],120,2))), IIF(MID([Field1],122,2)=' ',Null,MID([Field1],122,2))
FROM SEG04;
```

Query **mkVICTIM_CIRCUMSTANCE_Table** creates the VICTIM_CIRCUMSTANCE table structure. The table fields are named and data types are set when this query is executed. The relationship with table VICTIM is created by linking key fields UID and VIC_SEQ_NO with the same key fields in table VICTIM.

```
CREATE TABLE VICTIM_CIRCUMSTANCE (UID TEXT(21), VIC_SEQ_NO LONG, CIRCUM TEXT(2), CONSTRAINT fkVICTIMCIRCUMSTANCE FOREIGN KEY (UID, VIC_SEQ_NO) REFERENCES VICTIM)
```

Query **mkVICTIM_CIRCUMSTANCE_Index** creates the VICTIM_CIRCUMSTANCE table index. Fields UID and VIC_SEQ_NO are used as the index fields for table VICTIM_CIRCUMSTANCE.

```
CREATE INDEX VICTIM_CIRCUMSTANCE ON VICTIM_CIRCUMSTANCE(UID, VIC_SEQ_NO)
```

Queries **appVICTIM_CIRCUMSTANCE_Data01** through **appVICTIM_CIRCUMSTANCE_Data03** copy the data from fields UID, VIC_SEQ_NO, CIRCUM1, CIRCUM2 and ADD_CIRCUM in table VICTIM and loads the data into table VICTIM_CIRCUMSTANCE.

appVICTIM_CIRCUMSTANCE_Data01:

```
INSERT INTO VICTIM_CIRCUMSTANCE ( UID, VIC_SEQ_NO, CIRCUM )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].CIRCUM1
FROM VICTIM
WHERE CIRCUM1 IS NOT NULL;
```

appVICTIM_CIRCUMSTANCE_Data02:

```
INSERT INTO VICTIM_CIRCUMSTANCE ( UID, VIC_SEQ_NO, CIRCUM )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].CIRCUM2
FROM VICTIM
WHERE CIRCUM2 IS NOT NULL;
```

appVICTIM_CIRCUMSTANCE_Data03:

```
INSERT INTO VICTIM_CIRCUMSTANCE ( UID, VIC_SEQ_NO, CIRCUM )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].ADD_CIRCUM
FROM VICTIM
WHERE ADD_CIRCUM IS NOT NULL;
```

Query **mkVICTIM_INJURY_Table** creates the VICTIM_INJURY table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table VICTIM by linking key fields UID and VIC_SEQ_NO in table VICTIM_INJURY with the same key fields in table VICTIM.

```
CREATE TABLE VICTIM_INJURY(UID TEXT(21), VIC_SEQ_NO LONG, INJ_SEQ_NO LONG, INJ TEXT(1), CONSTRAINT fkVICTIMINJURY FOREIGN KEY (UID, VIC_SEQ_NO) REFERENCES VICTIM)
```

Query **mkVICTIM_INJURY_Index** creates the VICTIM_INJURY table index. Fields UID, VIC_SEQ_NO and INJ_SEQ_NO are used as the index fields for table VICTIM_INJURY.

```
CREATE INDEX VICTIM_INJURY ON VICTIM_INJURY(UID, VIC_SEQ_NO, INJ_SEQ_NO)
```

Queries **appVICTIM_INJURY_Data01** through **appVICTIM_INJURY_Data05** copy the data from fields UID, VIC_SEQ_NO, and INJ1 through INJ5 in table VICTIM and loads the data into table VICTIM_INJURY. The victim injury sequence number, INJ_SEQ_NO, is also loaded by these queries.

appVICTIM_INJURY_Data01:

```
INSERT INTO VICTIM_INJURY ( UID, VIC_SEQ_NO, INJ_SEQ_NO, INJ )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 1, VICTIM.INJ1
FROM VICTIM
WHERE INJ1 IS NOT NULL;
```

appVICTIM_INJURY_Data02:

```
INSERT INTO VICTIM_INJURY ( UID, VIC_SEQ_NO, INJ_SEQ_NO, INJ )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 2, VICTIM.INJ2
FROM VICTIM
WHERE INJ2 IS NOT NULL;
```

appVICTIM_INJURY_Data03:

```
INSERT INTO VICTIM_INJURY ( UID, VIC_SEQ_NO, INJ_SEQ_NO, INJ )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 3, VICTIM.INJ3
FROM VICTIM
WHERE INJ3 IS NOT NULL;
```

appVICTIM_INJURY_Data04:

```
INSERT INTO VICTIM_INJURY ( UID, VIC_SEQ_NO, INJ_SEQ_NO, INJ )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 4, VICTIM.INJ4
FROM VICTIM
WHERE INJ4 IS NOT NULL;
```

appVICTIM_INJURY_Data05:

```
INSERT INTO VICTIM_INJURY ( UID, VIC_SEQ_NO, INJ_SEQ_NO, INJ )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 5, VICTIM.INJ5
FROM VICTIM
WHERE INJ5 IS NOT NULL;
```

Query **mkVICTIM_OFFENSE_Table** creates the VICTIM_OFFENSE table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table VICTIM by linking key fields UID and VIC_SEQ_NO in table VICTIM_OFFENSE with the same key fields in table VICTIM.

```
CREATE TABLE VICTIM_OFFENSE(UID TEXT(21), VIC_SEQ_NO LONG, OFFNO LONG, OFF_CODE TEXT(3),
CONSTRAINT fkVICTIMOFFENSE FOREIGN KEY (UID, VIC_SEQ_NO) REFERENCES VICTIM)
```

Query **mkVICTIM_OFFENSE_Index** creates the VICTIM_OFFENSE table index. Fields UID, VIC_SEQ_NO and OFFNO are used as the index fields for table VICTIM_INJURY.

```
CREATE INDEX VICTIM_OFFENSE ON VICTIM_OFFENSE(UID, VIC_SEQ_NO, OFFNO)
```

Queries **appVICTIM_OFFENSE_Data01** through **appVICTIM_OFFENSE_Data10** copy the data from fields UID, VIC_SEQ_NO, and VICOFF1 through VICOFF10 in table VICTIM and loads the data into table VICTIM_OFFENSE. The victim offense sequence number, OFFNO, is loaded by these queries as well.

appVICTIM_OFFENSE_Data01:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 1, VICTIM.VICOFF1
FROM Victim
WHERE VICOFF1 IS NOT NULL;
```

appVICTIM_OFFENSE_Data02:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 2, VICTIM.VICOFF2
FROM Victim
WHERE VICOFF2 IS NOT NULL;
```

appVICTIM_OFFENSE_Data03:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 3, VICTIM.VICOFF3
FROM Victim
WHERE VICOFF3 IS NOT NULL;
```

appVICTIM_OFFENSE_Data04:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 4, VICTIM.VICOFF4
FROM Victim
WHERE VICOFF4 IS NOT NULL;
```

appVICTIM_OFFENSE_Data05:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 5, VICTIM.VICOFF5
```

```
FROM Victim
WHERE VICOFF5 IS NOT NULL;
```

appVICTIM_OFFENSE_Data06:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 6, VICTIM.VICOFF6
FROM Victim
WHERE VICOFF6 IS NOT NULL;
```

appVICTIM_OFFENSE_Data07:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 7, VICTIM.VICOFF7
FROM Victim
WHERE VICOFF7 IS NOT NULL;
```

appVICTIM_OFFENSE_Data08:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 8, VICTIM.VICOFF8
FROM Victim
WHERE VICOFF8 IS NOT NULL;
```

appVICTIM_OFFENSE_Data09:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 9, VICTIM.VICOFF9
FROM Victim
WHERE VICOFF9 IS NOT NULL;
```

appVICTIM_OFFENSE_Data10:

```
INSERT INTO VICTIM_OFFENSE ( UID, VIC_SEQ_NO, OFFNO, OFF_CODE )
SELECT VICTIM.UID, VICTIM.VIC_SEQ_NO, 10, VICTIM.VICOFF10
FROM Victim
WHERE VICOFF10 IS NOT NULL;
```

Query **mkVICTIM_TO_OFFENDER_Table** creates the VICTIM_TO_OFFENDER table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table VICTIM by linking key fields UID and VIC_SEQ_NO in table VICTIM_TO_OFFENDER with the same key fields in table VICTIM.

```
CREATE TABLE VICTIM_TO_OFFENDER (UID TEXT(21), VIC_SEQ_NO LONG, OFN_SEQ_NO LONG, VIC_2_OFN
TEXT(2), CONSTRAINT fkVICTIMTOOFFENDER FOREIGN KEY (UID, VIC_SEQ_NO) REFERENCES VICTIM)
```

Query **mkVICTIM_TO_OFFENDER_Index** creates the VICTIM_TO_OFFENDER table index. Fields UID, VIC_SEQ_NO and OFN_SEQ_NO are used as the index fields for table VICTIM_TO_OFFENDER.

```
CREATE INDEX VICTIM_TO_OFFENDER ON VICTIM_TO_OFFENDER(UID, VIC_SEQ_NO, OFN_SEQ_NO)
```

Queries **appVICTIM_TO_OFFENDER_Data01** through **appVICTIM_TO_OFFENDER_Data10** copy the data from fields UID, VIC_SEQ_NO, OFN1 through OFN10 and VOR1 through VOR10 in table VICTIM and loads the data into table VICTIM_TO_OFFENDER.

```
appVICTIM_TO_OFFENDER_Data01:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN1, [VICTIM].VOR1
FROM VICTIM
WHERE OFN1 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data02:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN2, [VICTIM].VOR2
FROM VICTIM
WHERE OFN2 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data03:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN3, [VICTIM].VOR3
FROM VICTIM
WHERE OFN3 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data04:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN4, [VICTIM].VOR4
FROM VICTIM
WHERE OFN4 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data05:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN5, [VICTIM].VOR5
FROM VICTIM
WHERE OFN5 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data06:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN6, [VICTIM].VOR6
FROM VICTIM
WHERE OFN6 IS NOT NULL;
```

```
appVICTIM_TO_OFFENDER_Data07:
```

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN7, [VICTIM].VOR7
FROM VICTIM
WHERE OFN7 IS NOT NULL;
```

appVICTIM_TO_OFFENDER_Data08:

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN8, [VICTIM].VOR8
FROM VICTIM
WHERE OFN8 IS NOT NULL;
```

appVICTIM_TO_OFFENDER_Data09:

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN9, [VICTIM].VOR9
FROM VICTIM
WHERE OFN9 IS NOT NULL;
```

appVICTIM_TO_OFFENDER_Data10:

```
INSERT INTO VICTIM_TO_OFFENDER ( UID, VIC_SEQ_NO, OFN_SEQ_NO, VIC_2_OFN )
SELECT [VICTIM].UID, [VICTIM].VIC_SEQ_NO, [VICTIM].OFN10, [VICTIM].VOR10
FROM VICTIM
WHERE OFN10 IS NOT NULL;
```

Query **mkDrop_Normalized_Fields_From_VICTIM** deletes from table VICTIM the fields of data now stored in tables VICTIM_CIRCUMSTANCE, VICTIM_INJURY, VICTIM_OFFENSE and VICTIM_TO_OFFENDER.

```
ALTER TABLE VICTIM DROP COLUMN VICOFF1, VICOFF2, VICOFF3, VICOFF4, VICOFF5, VICOFF6, VICOFF7,
VICOFF8, VICOFF9, VICOFF10, INJ1, INJ2, INJ3, INJ4, INJ5, OFN1, VOR1, OFN2, VOR2, OFN3, VOR3,
OFN4, VOR4, OFN5, VOR5, OFN6, VOR6, OFN7, VOR7, OFN8, VOR8, OFN9, VOR9, OFN10, VOR10, CIRCUM1,
CIRCUM2, ADD_CIRCUM
```

OFFENDER Segment

Query **mkOFFENDER_Table** creates the OFFENDER table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table ADMINISTRATIVE by linking key field UID in table OFFENDER with the same key fields in table ADMINISTRATIVE.

```
CREATE TABLE OFFENDER (SEG TEXT(2), UID TEXT(21), OFN_SEQ_NO LONG, OFN_AGE TEXT(2), OFN_SEX
TEXT(1), OFN_RACE TEXT(1), CONSTRAINT fkOFFENDERADMIN FOREIGN KEY(UID) REFERENCES
ADMINISTRATIVE)
```

Query **mkOFFENDER_Index** creates the OFFENDER table index. Fields UID and OFN_SEQ_NO are used as the index fields for table OFFENDER.

```
CREATE INDEX PrimaryKey ON OFFENDER(UID, OFN_SEQ_NO) WITH PRIMARY
```

Query **appOFFENDER_Data** is an append query that uses the NIBRS file record layout to format the data from segment-level file SEG05YY.txt and load it into table OFFENDER (SEG05YY.txt is known to the query as SEG01).

```
INSERT INTO OFFENDER ( SEG, UID, OFN_SEQ_NO, OFN_AGE, OFN_SEX, OFN_RACE )
SELECT IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)), IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(MID([Field1],34,2)=' ',Null,VAL(MID([Field1],34,2))), IIF(MID([Field1],36,2)=' ',Null,MID([Field1],36,2)),
IIF(Mid([Field1],38,1)=' ',Null,MID([Field1],38,1)), IIF(Mid([Field1],39,1)=' ',Null,MID([Field1],39,1))
FROM SEG05;
```

Query **mkOFFENDER_RELATIONSHIPS** is used to link the data from table OFFENDER to the offender data in table VICTIM_TO_OFFENDER. Fields UID and OFN_SEQ_NO are the key fields that appear in both tables. Linking these fields in the two tables creates the table relationship.

```
ALTER TABLE VICTIM_TO_OFFENDER ADD CONSTRAINT fkVICTIMOFFENDERTOOFFENDER FOREIGN
KEY(UID, OFN_SEQ_NO) REFERENCES OFFENDER
```

ARRESTEE Segment

Query **mkARRESTEEA_Table** creates the ARRESTEEA table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table ADMINISTRATIVE by linking key field UID in table ARRESTEEA with the same key fields in table ADMINISTRATIVE.

```
CREATE TABLE ARRESTEEA(SEG TEXT(2), UID TEXT(21), ARR_SEQ_NO LONG, ARR_DATE DATE, ARR_TYPE
TEXT(1), MULT_SEG TEXT(1), ARR_OFF_CODE TEXT(3), ARR_WEP1 TEXT(2), ARR_AUTO1 TEXT(1),
ARR_WEP2 TEXT(2), ARR_AUTO2 TEXT(1), ARR_AGE TEXT(2), ARR_SEX TEXT(1), ARR_RACE TEXT(1),
ARR_ETHN TEXT(1), ARR_RESID TEXT(1), ARR_DISP TEXT(1), CONSTRAINT fkARRESTEEAADMIN FOREIGN
KEY (UID) REFERENCES ADMINISTRATIVE)
```

Query **mkARRESTEE_Index** creates the ARRESTEE table index. Fields UID and ARR_SEQ_NO are used as the index fields for table ARRESTEEA.

```
CREATE INDEX PrimaryKey ON ARRESTEEA(UID, ARR_SEQ_NO) WITH PRIMARY
```

Query **appARRESTEEA_Data** is an append query that uses the NIBRS file record layout to format the data from segment-level file SEG06YY.txt and load it into table OFFENDER (SEG06YY.txt is known to the query as SEG06).

```

INSERT INTO ARRESTEEA ( SEG, UID, ARR_SEQ_NO, ARR_DATE, ARR_TYPE, MULT_SEG, ARR_OFF_CODE,
ARR_WEP1, ARR_AUTO1, ARR_WEP2, ARR_AUTO2, ARR_AGE, ARR_SEX, ARR_RACE, ARR_ETHN,
ARR_RESID, ARR_DISP )
SELECT IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)),
IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(MID([Field1],34,2)=' ',Null,VAL(MID([Field1],34,2))),
IIF(Mid([Field1],48,8)=' ',Null,DATEVALUE(MID([Field1],52,2) & '/' & MID([Field1],54,2) & '/' & MID([Field1],48,4))),
IIF(Mid([Field1],56,1)=' ',Null,MID([Field1],56,1)),IIF(Mid([Field1],57,1)=' ',Null,MID([Field1],57,1)),
IIF(MID([Field1],58,3)=' ',Null,MID([Field1],58,3)), IIF(MID([Field1],61,2)=' ',Null,MID([Field1],61,2)),
IIF(Mid([Field1],63,1)=' ',Null,MID([Field1],63,1)), IIF(MID([Field1],64,2)=' ',Null,MID([Field1],64,2)),
IIF(Mid([Field1],66,1)=' ',Null,MID([Field1],66,1)), IIF(Mid([Field1],67,2)=' ',Null,MID([Field1],67,2)),
IIF(Mid([Field1],69,1)=' ',Null,MID([Field1],69,1)), IIF(Mid([Field1],70,1)=' ',Null,MID([Field1],70,1)),
IIF(Mid([Field1],71,1)=' ',Null,MID([Field1],71,1)), IIF(Mid([Field1],72,1)=' ',Null,MID([Field1],72,1)),
IIF(Mid([Field1],73,1)=' ',Null,MID([Field1],73,1))
FROM SEG06

```

Query **mkARRESTEEA_WEAPON_Table** creates the ARRESTEEA_WEAPON table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table ARRESTEEA by linking key fields UID and ARR_SEQ_NO in table ARRESTEEA_WEAPON with the same key fields in table ARRESTEEA.

```

CREATE TABLE ARRESTEEA_WEAPON(UID TEXT(21), ARR_SEQ_NO LONG, ARR_OFF_CODE TEXT(3),
AWEP_SEQ_NO LONG, WEP TEXT(2), AUTO TEXT(1), CONSTRAINT fkARRESTEEAWEAPON FOREIGN KEY
(UID, ARR_SEQ_NO) REFERENCES ARRESTEEA)

```

Query **mkARRESTEEA_WEAPON_Index** creates the ARRESTEEA_WEAPON table index. Fields UID, ARR_SEQ_NO and AWEP_SEQ_NO are used as the index fields for table ARRESTEEA_WEAPON.

```

CREATE INDEX ARRESTEEA_WEAPON ON ARRESTEEA_WEAPON(UID, ARR_SEQ_NO, AWEP_SEQ_NO)

```

Queries **appARRESTEEA_WEAPON_Data01** and **appARRESTEEA_WEAPON_Data02** copy the data from fields UID, ARR_SEQ_NO, ARR_OFF_CODE, WEP1 and WEP2 and ARR_AUTO1 and ARR_AUTO2 in table ARRESTEEA and loads the data into table ARRESTEEA_WEAPON. The arrestee weapon sequence number, AWEP_SEQ_NO, is also loaded by these queries.

appARRESTEEA_WEAPON_Data01:

```
INSERT INTO ARRESTEEA_WEAPON ( UID, ARR_SEQ_NO, ARR_OFF_CODE, AWEP_SEQ_NO, WEP, AUTO )
SELECT [ARRESTEEA].[UID], [ARRESTEEA].[ARR_SEQ_NO], [ARRESTEEA].[ARR_OFF_CODE], 1,
[ARRESTEEA].ARR_WEP1, [ARRESTEEA].ARR_AUTO1
FROM ARRESTEEA
WHERE [ARRESTEEA].ARR_WEP1 Is Not Null;
```

appARRESTEEA_WEAPON_Data02:

```
INSERT INTO ARRESTEEA_WEAPON ( UID, ARR_SEQ_NO, ARR_OFF_CODE, AWEP_SEQ_NO, WEP, AUTO )
SELECT [ARRESTEEA].[UID], [ARRESTEEA].[ARR_SEQ_NO], [ARRESTEEA].ARR_OFF_CODE, 2,
[ARRESTEEA].ARR_WEP2, [ARRESTEEA].ARR_AUTO2
FROM ARRESTEEA
WHERE [ARRESTEEA].ARR_WEP2 Is Not Null;
```

Query **mkDrop_Normalized_Fields_From_ARRESTEEA** deletes from table ARRESTEEA the fields of data now stored in table ARRESTEEA_WEAPON.

```
mkDrop_Normalized_Fields_From_ARRESTEEA
ALTER TABLE ARRESTEEA DROP COLUMN ARR_WEP1, ARR_AUTO1, ARR_WEP2, ARR_AUTO2
```

ARRESTEE B Segment

Query **mkARRESTEEB_Table** creates the ARRESTEEB table structure. The table fields are named and data types are set when this query is executed.

```
CREATE TABLE ARRESTEEB(SEG TEXT(2), ARRB_UID TEXT(21), ARR_DATE DATE, ARR_SEQ_NO LONG,
CITY_SUB TEXT(4), ARR_TYPE TEXT(1), ARR_OFF_CODE TEXT(3), ARR_WEP1 TEXT(2), ARR_AUTO1 TEXT(1),
ARR_WEP2 TEXT(2), ARR_AUTO2 TEXT(1), ARR_AGE TEXT(2), ARR_SEX TEXT(1), ARR_RACE TEXT(1),
ARR_ETHN TEXT(1), ARR_RESID TEXT(1), ARR_DISP TEXT(1))
```

Query **mkARRESTEB_Index** creates the ARRESTEB table index. Fields ARRB_UID and ARR_SEQ_NO are used as the index fields for table ARRESTEEB. The query also establishes these same fields as the key fields for table ARRESTEEB.

```
CREATE INDEX PrimaryKey ON ARRESTEEB(ARRB_UID, ARR_SEQ_NO) WITH PRIMARY
```

Query **appARRESTEEB_Data** is an append query that uses the NIBRS file record layout to format the data from segment-level file SEG07YY.txt and load it into table OFFENDER (SEG07YY.txt is known to the query as SEG07).

```

INSERT INTO ARRESTEEB ( SEG, ARRB_UID, ARR_DATE, ARR_SEQ_NO, CITY_SUB, ARR_TYPE,
ARR_OFF_CODE, ARR_WEP1, ARR_AUTO1, ARR_WEP2, ARR_AUTO2, ARR_AGE, ARR_SEX, ARR_RACE,
ARR_ETHN, ARR_RESID, ARR_DISP )
SELECT
IIF(LEFT([Field1],2)=' ',Null,LEFT([Field1],2)), IIF(MID([Field1],5,21)=' ',Null,MID([Field1],5,21)),
IIF(Mid([Field1],26,8)=' ',Null,DATEVALUE(MID([Field1],30,2) & '/' & MID([Field1],32,2) & '/' & MID([Field1],26,4))),
IIF(MID([Field1],34,2)=' ',Null,VAL(MID([Field1],34,2))), IIF(Mid([Field1],36,4)=' ',Null,MID([Field1],36,4)),
IIF(Mid([Field1],40,1)=' ',Null,MID([Field1],40,1)), IIF(MID([Field1],41,3)=' ',Null,MID([Field1],41,3)),
IIF(MID([Field1],44,2)=' ',Null,MID([Field1],44,2)), IIF(Mid([Field1],46,1)=' ',Null,MID([Field1],46,1)),
IIF(MID([Field1],47,2)=' ',Null,MID([Field1],47,2)), IIF(Mid([Field1],49,1)=' ',Null,MID([Field1],49,1)),
IIF(Mid([Field1],50,2)=' ',Null,VAL(MID([Field1],50,2))), IIF(Mid([Field1],52,1)=' ',Null,MID([Field1],52,1)),
IIF(Mid([Field1],53,1)=' ',Null,MID([Field1],53,1)), IIF(Mid([Field1],54,1)=' ',Null,MID([Field1],54,1)),
IIF(Mid([Field1],55,1)=' ',Null,MID([Field1],55,1)), IIF(Mid([Field1],56,1)=' ',Null,MID([Field1],56,1))
FROM SEG07;

```

Query **mkARRESTEEB_WEAPON_Table** creates the ARRESTEEB_WEAPON table structure. The table fields are named and data types are set when this query is executed. The relationship is created with table ARRESTEEB by linking key fields UID and ARR_SEQ_NO in table ARRESTEEB_WEAPON with the same key fields in table ARRESTEEB.

```

CREATE TABLE ARRESTEEB_WEAPON(ARRB_UID TEXT(21), ARR_SEQ_NO LONG, ARR_OFF_CODE TEXT(3),
BWEP_SEQ_NO LONG, WEP TEXT(2), AUTO TEXT(1), CONSTRAINT fkARRESTEEB_WEAPON FOREIGN KEY
(ARRB_UID, ARR_SEQ_NO) REFERENCES ARRESTEEB)

```

Query **mkARRESTEEB_WEAPON_Index** creates the ARRESTEEB_WEAPON table index. Fields ARRB_UID, ARR_SEQ_NO, ARR_OFF_CODE and BWEP_SEQ_NO are used as the index fields for table ARRESTEEB_WEAPON.

```

CREATE INDEX ARRESTEEB_WEAPON ON ARRESTEEB_WEAPON(ARRB_UID, ARR_SEQ_NO,
ARR_OFF_CODE, BWEP_SEQ_NO)

```

Queries **appARRESTEEB_WEAPON_Data01** and **appARRESTEEB_WEAPON_Data02** copy the data from fields ARRB_UID, ARR_SEQ_NO, ARR_OFF_CODE, ARR_WEP1 and ARR_WEP2 and ARR_AUTO1 and ARR_AUTO2 in table ARRESTEEB and loads the data into table ARRESTEEB_WEAPON. The arrestee weapon sequence number, BWEP_SEQ_NO, is also loaded by these queries.

appARRESTEEB_WEAPON_Data01

```
INSERT INTO ARRESTEEB_WEAPON ( ARRB_UID, ARR_SEQ_NO, ARR_OFF_CODE, BWEP_SEQ_NO, WEP,  
AUTO )  
SELECT [ARRESTEEB].ARRB_UID, [ARRESTEEB].ARR_SEQ_NO, [ARRESTEEB].ARR_OFF_CODE, 1,  
[ARRESTEEB].ARR_WEP1, [ARRESTEEB].ARR_AUTO1  
FROM ARRESTEEB  
WHERE [ARRESTEEB].ARR_WEP1 Is Not Null;
```

appARRESTEEB_WEAPON_Data02

```
INSERT INTO ARRESTEEB_WEAPON ( ARRB_UID, ARR_SEQ_NO, ARR_OFF_CODE, BWEP_SEQ_NO, WEP,  
AUTO )  
SELECT [ARRESTEEB].ARRB_UID, [ARRESTEEB].ARR_SEQ_NO, [ARRESTEEB].ARR_OFF_CODE, 2,  
[ARRESTEEB].ARR_WEP2, [ARRESTEEB].ARR_AUTO2  
FROM ARRESTEEB  
WHERE [ARRESTEEB].ARR_WEP2 Is Not Null;
```

Query `mkDrop_Normalized_Fields_From_ARRESTEEB` deletes from table `ARRESTEEB` the fields of data now stored in table `ARRESTEEB_WEAPON`.

```
ALTER TABLE ARRESTEEB DROP COLUMN ARR_WEP1, ARR_AUTO1, ARR_WEP2, ARR_AUTO2
```