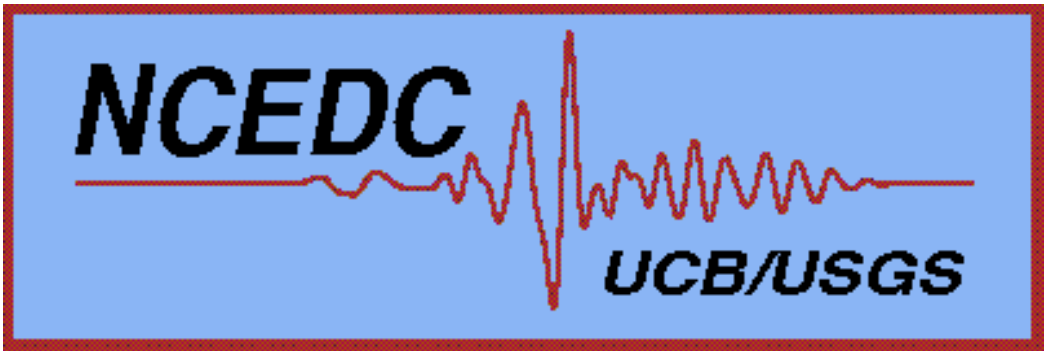
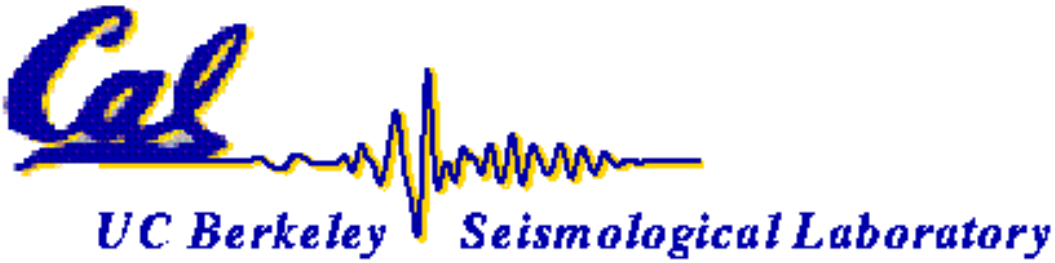



# parametric information schema



# Table of Contents

**1.- Introduction.** ..... 2

**2.- Database structure.** ..... 3

**3.- Database relations.** ..... 12

**4.- Database attributes.** ..... 18

**5.- Primary keys.** ..... 73

**6.- Time issues.** ..... 74

## ***1.- Introduction :***

This paper is about the design of the NCEDC schema for the parametric information. This schema was designed to be able to store all the information from the CNSS format, following as much as possible the CSS 3.0 schema.

There are 14 relations in the schema. The information stored in these relations includes:

- Event information, including preferred origin, preferred magnitude, etc.
- Multiple origins for an event.
- Multiple magnitudes for each origin.
- Arrivals (seismic signals).
- Association of arrivals with origins.
- Association of amplitudes with origins, as well as with magnitudes.
- Moment tensor information.
- Association of coda information with magnitudes.
- Association of coda information with origins.

The remainder of this volume consists of five chapters:

| | |
|---|---|
| Chapter 2 | *Database structure*<br>Each relation is defined. |
| Chapter 3 | *Database relations*<br>The logical design of the database is expressed in an Entity-Relationship diagram and each relation is described to identify the key fields and the links among the relations. |
| Chapter 4 | *Database attributes*<br>Each attribute is described. |
| Chapter 5 | *Primary keys*<br>The notion of sequence is described. |
| Chapter 6 | *Time issues*<br>A *PL/SQL* implementation of time conversion is given. |

## *2.- Database structure :*

This chapter defines the physical structure of each table, as it exists within the DBMS data dictionary. The name of the relation appears in **bold** print at the top left and a description of the table at the top right. **Bold** attributes represent primary keys, *italic* attributes represent foreign keys, and underlined attributes stand for CSS attributes.

| **Event** | | Summary information of an event |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **evid** | int | Event identifier |
| *prefor* | int | Preferred origin |
| *prefmag* | int | Preferred magnitude |
| *prefmec* | int | Preferred mechanism |
| *commid* | int | Comment identifier |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| totalarr | int | Total number of arrivals |
| totalamp | int | Total number of amplitude readings |
| etype | char(7) | Event type |
| lddate | date | Load date |

| **Significant_Event** | | Summary information of a significant event |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **evid** | int | Event identifier |
| evname | char(80) | Event name |
| remarks | char(2) | Event indications (felt, damage, ...) |
| nfelt | int | Number of felt reports |
| mmi | int | Maximum MM intensity |
| pga | float | Peak acceleration |
| lddate | date | Load date |

| **Origin** | | Data on event location |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **orid** | int | Origin identifier |
| *evid* | int | Event identifier |
| *prefmag* | int | Preferred magnitude |
| *prefmec* | int | Preferred mechanism |
| *commid* | int | Comment identifier |
| datetime | float | Origin date |
| lat | float | Estimated latitude |
| lon | float | Estimated longitude |
| depth | float | Estimated depth |

| type | char(2) | Type of location |
|---|---|---|
| algorithm | char(15) | Location algorithm used |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| datumhor | char(8) | Datum type for latitude & longitude |
| datumver | char(8) | Datum type for vertical |
| gap | float | Maximum azimuthal gap of stations |
| distance | float | Distance to nearest station |
| sdobs | float | Standard error of observations |
| stime | float | Origin time error |
| erhor | float | Horizontal error |
| sdepth | float | Depth error |
| erlat | float | Error in latitude |
| erlon | float | Error in longitude |
| ndef | int | Number of locating phases |
| nbs | int | Number of S readings |
| nbp | int | Number of P first motions |
| sxx | float | Covariance matrix element |
| syy | float | Covariance matrix element |
| szz | float | Covariance matrix element |
| stt | float | Covariance matrix element |
| sxy | float | Covariance matrix element |
| sxz | float | Covariance matrix element |
| syz | float | Covariance matrix element |
| stx | float | Covariance matrix element |
| sty | float | Covariance matrix element |
| stz | float | Covariance matrix element |
| azismall | float | Azimuth of smallest principal error |
| dipsmall | float | Dip of smallest principal error |
| magsmall | float | Magnitude of smallest principal error |
| aziinter | float | Azimuth of intermediate principal error |
| dipinter | float | Dip of intermediate principal error |
| maginter | float | Magnitude of intermediate principal error |
| azilarge | float | Azimuth of largest principal error |
| diplarge | float | Dip of largest principal error |
| maglarge | float | Magnitude of largest principal error |
| locevid | int | Local event identifier |
| quality | float | Quality of solution |
| fdepth | char | Fixed depth |
| fepi | char | Fixed epicenter |
| ftime | char | Fixed time |
| vmodelid | int | Velocity model identifier |
| cmodelid | int | Crustal model identifier |
| lddate | date | Load date |

| **Netmag** | | Summary information on a magnitude |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **magid** | int | Magnitude identifier |
| *orid* | int | Origin identifier |
| *evid* | int | Event identifier |
| *commid* | int | Comment identifier |
| magnitude | float | Magnitude |
| magtype | char(6) | Magnitude type |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| magalgo | char(15) | Magnitude algorithm |
| nsta | int | Number of stations used |
| uncertainty | float | Magnitude uncertainty |
| gap | float | Maximum azimuthal gap of stations |
| distance | float | Minimum distance of stations |
| lddate | date | Load date |

| **Arrival** | | Summary information on a seismic arrival |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **arid** | int | Arrival identifier |
| *evid* | int | Event identifier |
| *commid* | int | Comment identifier |
| datetime | float | Pick date |
| sta | char(6) | Station code |
| net | char(8) | Network identifier |
| iphase | char(8) | Reported phase |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| channel | char(3) | Channel name |
| channelsrc | char(8) | Domain of channel name |
| seedchan | char(3) | SEED channel name |
| qual | char(1) | Signal onset quality |
| fm | char(2) | First motion |
| ema | float | Emergence angle |
| azimuth | float | Observed azimuth |
| slow | float | Observed slowness |
| deltim | float | Delta time |
| delinc | float | Error in angle |
| delaz | float | Delta azimuth |
| delslo | float | Delta slowness |
| quality | float | Quality |
| snr | float | Signal to noise ratio |
| lddate | date | Load date |

| **AssocArO** | | Data associating arrivals with origins |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **orid** | int | Origin identifier |
| **arid** | int | Arrival identifier |
| *commid* | int | Comment identifier |
| iphase | char(8) | Associated phase |
| importance | float | Importance of phase |
| delta | float | Station to event distance |
| seaz | float | Station to event azimuth |
| wgt | float | Location weight |
| timeres | float | Time residual |
| azres | float | Azimuth residual |
| emares | float | Incidence angle residual |
| slores | float | Slowness residual |
| vmodelid | int | Velocity model identifier |
| scorr | float | Station correction in seconds |
| sdelay | float | Station delay |
| lddate | date | Load date |

| **Amp** | | Summary information on an amplitude reading |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **ampid** | int | Amplitude identifier |
| *evid* | int | Event identifier |
| *commid* | int | Comment identifier |
| datetime | float | Amplitude date |
| sta | char(6) | Station code |
| net | char(8) | Network identifier |
| iphase | char(8) | Reported phase |
| amplitude | float | Amplitude |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| channel | char(3) | Channel name |
| channelsrc | char(8) | Source of channel name |
| seedchan | char(3) | SEED channel name |
| amptype | char(3) | Amplitude type |
| units | char(4) | Units of amplitude |
| ampmeas | char(1) | Amplitude measure |
| eramp | float | Error in amplitude |
| flagamp | char(4) | Flag to indicate whether amplitude is over P packet,S packet,entire waveform. |
| per | float | Period |
| snr | float | Signal to noise ratio |
| tau | float | Coda duration (F-P time) |
| durtype | char(3) | Duration type |
| lddate | date | Load date |

| **AssocAmO** | | Data associating amplitudes with origins |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **orid** | int | Origin identifier |
| **ampid** | int | Amplitude identifier |
| *commid* | int | Comment identifier |
| delta | float | Station to event distance |
| seaz | float | Station to event azimuth |
| importance | float | Importance of amplitude |
| lddate | date | Load date |

| **AssocAmM** | | Data associating amplitudes with magnitudes |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **magid** | int | Magnitude identifier |
| **ampid** | int | Amplitude identifier |
| *commid* | int | Comment identifier |
| weight | float | Weight code |
| mag | float | Magnitude for this station reading |
| magres | float | Magnitude residual |
| magcorr | float | Magnitude correction for this station |
| lddate | date | Load date |

| **Remark** | | Comments |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **commid** | int | Comment identifier |
| **lineno** | int | Comment line number |
| remark | char(80) | Free format comment |
| lddate | date | Load date |

| **Mec** | | Summary information on a moment tensor |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **mecid** | int | Mechanism identifier |
| *oridin* | int | Origin identifier (input) |
| *oridout* | int | Origin identifier (output) |
| *evid* | int | Event identifier |
| *magid* | int | Mw magnitude identifier |
| *commid* | int | Comment identifier |
| mechtype | char(2) | Mechanism/moment type |
| mecalgo | char(15) | Mechanism algorithm |
| scalar | float | Scalar moment |
| erscalar | float | Error in scalar moment |
| tft | char(8) | Time function type |
| tfd | float | Time function duration |
| exp | int | Exponent of moment tensor elements |
| mxx | float | m_xx |
| myy | float | m_yy |
| mzz | float | m_zz |
| mxy | float | m_xy |
| mxz | float | m_xz |
| myz | float | m_yz |
| unmxx | float | Uncertainty in m_xx |
| unmyy | float | Uncertainty in m_yy |
| unmzz | float | Uncertainty in m_zz |

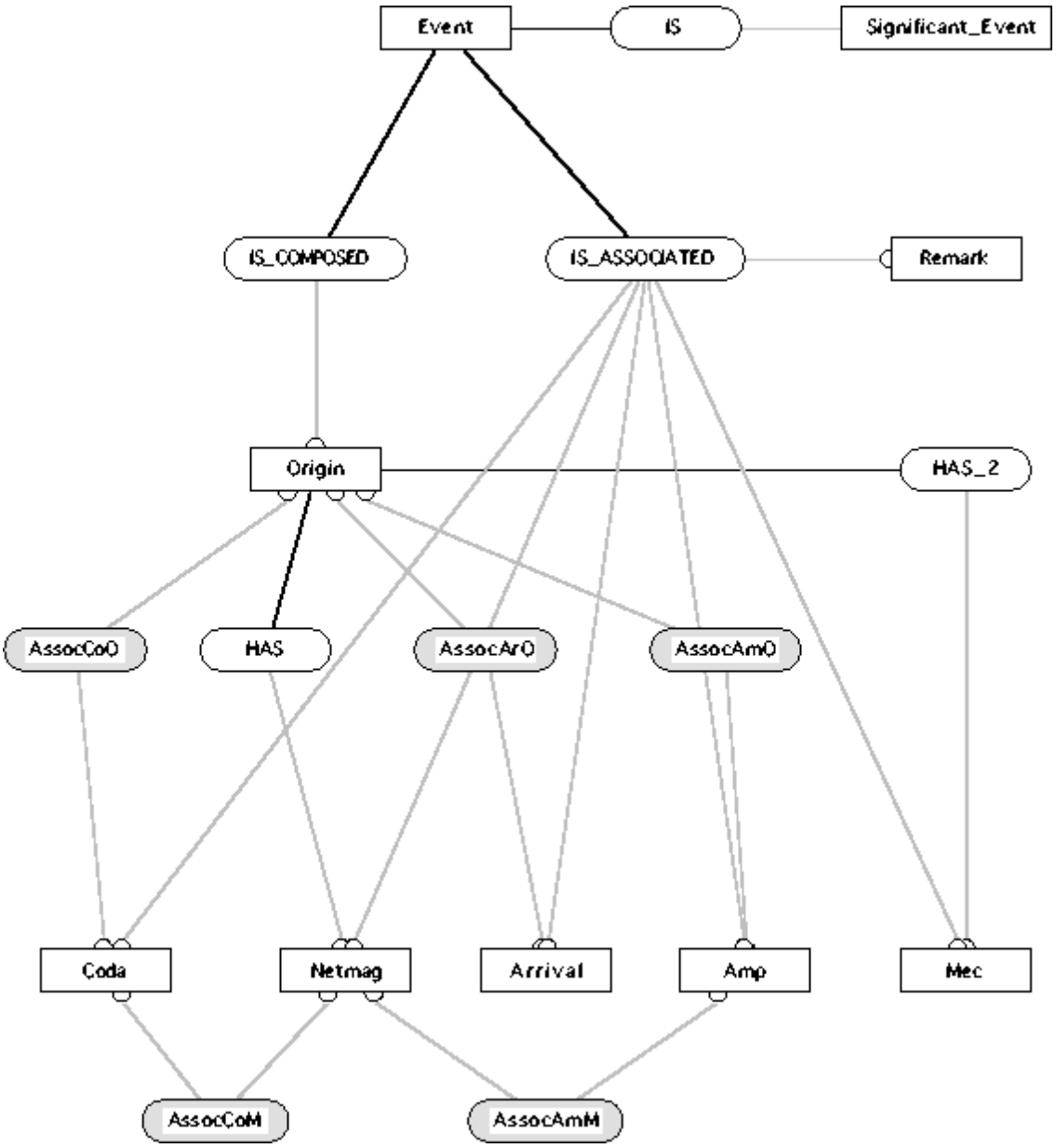
| | | |
|---|---|---|
| unmxy | float | Uncertainty in m_xy |
| unmxz | float | Uncertainty in m_xz |
| unmyz | float | Uncertainty in m_yz |
| srcduration | float | Source half-duration |
| auth | char(15) | Source/origin |
| subsource | char(8) | Subsystem source |
| strike1 | int | Strike 1 of best double couple |
| dip1 | int | Dip 1 of best double couple |
| rake1 | int | Rake 1 of best double couple |
| strike2 | int | Strike 2 of best double couple |
| dip2 | int | Dip 2 of best double couple |
| rake2 | int | Rake 2 of best double couple |
| unstrike1 | float | Uncertainty in strike 1 |
| undip1 | float | Uncertainty in dip 1 |
| unrake1 | float | Uncertainty in rake 1 |
| unstrike2 | float | Uncertainty in strike 2 |
| undip2 | float | Uncertainty in dip 2 |
| unrake2 | float | Uncertainty in rake 2 |
| eigenp | float | Eigen pressure |
| plungep | int | Plunge pressure |
| strikep | int | Strike pressure |
| eigenn | float | Eigen null |
| plungen | int | Plunge null |
| striken | int | Strike null |
| eigent | float | Eigen tension |
| plunget | int | Plunge tension |
| striket | int | Strike tension |
| nsta | int | Number of stations used in solution |
| pvr | int | Percent variance reduction |
| quality | float | Quality |
| pdc | int | Percent double couple |
| pclvd | int | Percent compensated linear vector dipole |
| piso | int | Percent isotropic |
| datetime | float | Date solution created |
| lddate | date | Load date |

| **Coda** | | Summary information on a coda |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **coid** | int | Coda identifier |
| *evid* | int | Event identifier |
| *commid* | int | Comment identifier |
| sta | char(6) | Station code |
| net | char(8) | Network code |
| auth | char(15) | Source/originator |
| subsource | char(8) | Subsystem source |
| channel | char(3) | Channel name |
| channelsrc | char(8) | Source of channel name |
| seedchan | char(3) | SEED channel name |
| codatype | char(3) | Coda type |
| afix | float | Nominal coda amplitude |
| afree | float | Free amplitude |
| qfix | float | Fixed coda decay constant |
| qfree | float | Free decay |
| tau | float | Coda duration (F-P time) |
| nsample | int | Number of code sample windows |
| rms | float | Residual of mean squares for coda fit |
| durtype | char(3) | Duration type |
| iphase | char(8) | Reported phase |
| eramp | float | Error in amplitude |
| units | char(4) | Units of amplitude |
| time1 | int | Time for pair 1 |
| amp1 | int | Amplitude for pair 1 |
| time2 | int | Time for pair 2 |
| amp2 | int | Amplitude for pair 2 |
| time3 | int | Time for pair 3 |
| amp3 | int | Amplitude for pair 3 |
| time4 | int | Time for pair 4 |
| amp4 | int | Amplitude for pair 4 |
| time5 | int | Time for pair 5 |
| amp5 | int | Amplitude for pair 5 |
| time6 | int | Time for pair 6 |
| amp6 | int | Amplitude for pair 6 |
| lddate | date | Load date |

| **AssocCoM** | | Data associating codas with magnitudes |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **magid** | int | Magnitude identifier |
| **coid** | int | Coda identifier |
| *commid* | int | Comment identifier |
| lddate | date | Load date |

| **AssocCoO** | | Data associating codas with origins |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| **orid** | int | Origin identifier |
| **coid** | int | Coda identifier |
| *commid* | int | Comment identifier |
| lddate | date | Load date |

***3.- Database relations :***



This chapter describes the DBMS relations that comprise the version 1.4.1 schema. The information given here, along with that in Chapter 4, *Database attributes*, constitutes the data dictionary. There's an entry for each relation. Within the entry, the relation's name appears first, followed by a list of its attributes. A brief description completes the entry.

Keys provide the link by which tables are joined. The following definitions explain the several types of keys.

- A primary key (which can be a concatenation of several attributes) uniquely identifies a row in the table. For example, each **Origin** record is unique by *lat, lon, depth,* and *time*.
- An alternate key also uniquely identifies a row in the table and may be used as the primary key. For example, *orid* may also be used as the primary key for the **Origin** table.
- A foreign key is another table's primary key. Thus, *evid* is a foreign key in the **Origin** table, but is the primary key in the **Event** table. Similarly, *commid* is a foreign key in many of the tables and the primary key in **Remark**.

| | |
|---|---|
| Name: | **Event** |
| Keys: | |
| | Primary: *evid* |
| | Foreign: *prefor, prefmag, prefmec, commid* |
| Data: | *auth, subsource, totalarr, totalamp, etype, lddate* |
| Description: | Event to origin connection. The purpose of this relation is to allow the connection of multiple origins to one event.<br>*prefor* points to the preferred origin.<br>*prefmag* points to the preferred magnitude.<br>*prefmec* points to the preferred mechanism. |

| | |
|---|---|
| Name: | **Significant_Event** |
| Keys: | |
| | Primary: *evid* |
| Data: | *evname, remarks, nfelt, mmi, pga, lddate* |
| Description: | This relation stores information about significant events. |

| | |
|---|---|
| Name: | **Origin** |
| Keys: | |
| | Primary: *orid*<br>Foreign: *evid, prefmag, prefmec, commid* |
| Data: | *datetime, lat, lon depth, type, algorithm, auth, subsource, datumhor, datumver, gap, distance, sdobs, stime, erhor, sdepth, erlat, erlon, ndef, nbs, nbp, sxx, syy, szz, stt, sxy, sxz, syz, stx, sty, stz, azismall, dipsmall, magsmall, aziinter, dipinter, maginter, azilarge, diplarge, maglarge, locevid, quality, fdepth, fepi, ftime, vmodelid, cmodelid, lddate* |
| Description: | Summary of hypocentral parameters. Information describing a derived or reported origin for a particular event is stored in this table. |

| | |
|---|---|
| Name: | **Netmag** |
| Keys: | |
| | Primary: *magid*<br>Foreign: *orid, evid, commid* |
| Data: | *magnitude, magtype, auth, subsource, magalgo, nsta, uncertainty, gap, distance, lddate* |
| Description: | This table summarizes estimates of magnitudes of different types for an event. Each magnitude has a unique *magid*. |

| | |
|---|---|
| Name: | **Arrival** |
| Keys: | |
| | Primary: *arid*<br>Foreign: *evid, commid* |
| Data: | *datetime, sta, net, iphase, auth, subsource, channel, channelsrc, seedchan, qual, fm, ema, azimuth, slow, deltim, delinc, delaz, delslo, quality, snr, lddate* |
| Description: | Summary information on a seismic arrival. Information characterizing a "seismic phase" observed at a particular station is saved here. |

| | |
|---|---|
| Name: | **AssocArO** |
| Keys: | |
| | Primary: *orid, arid*<br>Foreign: *commid* |
| Data: | *iphase, importance, delta, seaz, wgt, timeres, azres, emares, slores, vmodelid, scorr, sdelay, lddate* |
| Description: | Data associating arrivals with origins. This table has information that connects arrivals (i.e., entries in the **Arrival** relation) to a particular origin. It has a composite key made of *arid* and *orid*. |

| | |
|---|---|
| Name: | **Amp** |
| Keys: | |
| | Primary: *ampid*<br>Foreign: *evid, commid* |
| Data: | *datetime, sta, net, iphase, amplitude, auth, subsource, channel, channelsrc, seedchan, amptype, units, ampmeas, eramp, flagamp, per, snr, tau, durtype, lddate* |
| Description: | This table contains information about amplitude readings. |

| | |
|---|---|
| Name: | **AssocAmO** |
| Keys: | |
| | Primary: *orid, ampid*<br>Foreign: *commid* |
| Data: | *delta, seaz, importance, lddate* |
| Description: | Data associating amplitudes with origins. This table has information that connects amplitudes (i.e., entries in the **Amp** relation) to a particular origin. It has a composite key made of *ampid* and *orid*. |

| | |
|---|---|
| Name: | **AssocAmM** |
| Keys: | |
| | Primary: *magid, ampid*<br>Foreign: *commid* |
| Data: | *weight, mag, magres, magcorr, lddate* |
| Description: | Data associating amplitudes with magnitudes. This table has information that connects amplitudes (i.e., entries in the **Amp** relation) to a particular magnitude. It has a composite key made of *magid* and *ampid*. |

| | |
|---|---|
| Name: | **Remark** |
| Keys: | |
| | Primary: *commid*, *lineno* |
| Data: | *remark, lddate* |
| Description: | Comments. This relation may be used to store free-form comments that embellish records of other relations. The *commid* field in many relations refers to a tuple in the **Remark** table. |

| | |
|---|---|
| Name: | **Mec** |
| Keys: | |
| | Primary: *mecid*<br>Foreign: *evid, magid, commid* |
| Data: | *oridin, oridout, mechtype, mecalgo, scalar, erscalar, tft, tfd, exp, mxx, myy, mzz, mxy, mxz, myz, unmxx, unmyy, unmzz, unmxy, unmxz, unmyz, srcduration, auth, subsource, strike1, dip1, rake1, strike2, dip2, rake2, unstrike1, undip1, unrake1, unstrike2, undip2, unrake2, eigenp, plungep, strikep, eigenn, plungen, striken, eigent, plunget, striket, nsta, pvr, quality, pdc, pclvd, piso, datetime, lddate* |
| Description: | Mechanisms. This relation is used to store information about moment tensors. |

| | |
|---|---|
| Name: | **Coda** |
| Keys: | |
| | Primary: *coid*<br>Foreign: *evid, commid* |
| Data: | *sta, net, auth, subsource, channel, channelsrc, seedchan, codatype, afix, afree, qfix, qfree, tau, nsample, rms, durtype, iphase, eramp, units, time1, amp1, time2, amp2, time3, amp3, time4, amp4, time5, amp5, time6, amp6, lddate* |
| Description: | This relation is used to store information about codas. |

| | |
|---|---|
| Name: | **AssocCoM** |
| Keys: | |
| | Primary: *magid, coid*<br>Foreign: *commid* |
| Data: | *lddate* |
| Description: | Data associating codas with magnitudes. This table has information that connects codas (i.e., entries in the **Coda** relation) to a particular magnitude. It has a composite key made of *magid* and *coid*. |

| | |
|---|---|
| Name: | **AssocCoO** |
| Keys: | |
| | Primary: *orid, coid*<br>Foreign: *commid* |
| Data: | *lddate* |
| Description: | Data associating codas with origins. This table has information that connects codas (i.e., entries in the **Coda** relation) to a particular origin. It has a composite key made of *orid* and *coid*. |

## ***4.- Database attributes :***

This chapter describes each of the attributes used in the version 1.4.1 schema. Descriptions of the relations are found in Chapter 3, *Database relations*.

| | |
|---|---|
| Name: | *afix* |
| Relations: | **Coda** |
| Description: | Nominal coda amplitude. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *afix* > 0.0 |

| | |
|---|---|
| Name: | *afree* |
| Relations: | **Coda** |
| Description: | Free amplitude. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *afree* > 0.0 |

| | |
|---|---|
| Name: | *algorithm* |
| Relations: | **Origin** |
| Description: | Location algorithm used. This is a brief textual description of the algorithm used for computing a seismic origin. |
| Datatype: | ***char(15)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 15 characters |

| | |
|---|---|
| Name: | *ampid* |
| Relations: | **Amp**, **AssocAmO**, **AssocAmM** |
| Description: | Amplitude identification. Each amplitude is assigned a unique positive integer identifying it. This number is used in the **AssocAmO** relation along with the origin identifier to link **Amp** and **Origin**. This number is used also in the **AssocAmM** relation along with the magnitude identifier to link **Amp** and **Netmag**. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *ampid* > 0 |

| | |
|---|---|
| Name: | *amplitude* |
| Relations: | **Amp** |
| Description: | Amplitude in appropriate units for type. |
| Datatype: | ***float*** |
| NULL Value: | ***No*** |
| Range: | *amplitude* > 0.0 |

| | |
|---|---|
| Name: | *ampmeas* |
| Relations: | **Amp** |
| Description: | Amplitude measure. |
| Datatype: | ***char(1)*** |
| NULL Value: | ***Yes*** |
| Range: | *ampmeas* ∈ {0 \| 1}<br>0 = peak-to-peak<br>1 = zero-to-peak |

| | |
|---|---|
| Name: | *amptype* |
| Relations: | **Amp** |
| Description: | Amplitude type. |
| Datatype: | ***char(3)*** |
| NULL Value: | ***Yes*** |
| Range: | *amptype* ∈ {C \| WA \| WAS \| PGA \| PGV \| PGD}<br>C = coda<br>WA = Wood-Anderson photographic<br>WAS = Wood-Anderson synthetic<br>PGA = peak ground acceleration<br>PGV = peak ground velocity<br>PGD = peak ground displacement |

| | |
|---|---|
| Name: | *amp1, amp2, amp3, amp4, amp5, amp6* |
| Relations: | ***Coda*** |
| Description: | Amplitude for pairs. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *amp1, amp2, amp3, amp4, amp5, amp6* > 0 |

| | |
|---|---|
| Name: | *arid* |
| Relations: | **Arrival**, **AssocArO** |
| Description: | Arrival identifier. Each arrival is assigned a unique positive integer identifying it. This number is used in the **AssocArO** relation along with the origin identifier to link **Arrival** and **Origin**. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *arid* > 0 |

| | |
|---|---|
| Name: | *auth* |
| Relations: | **Event**, **Origin**, **Netmag**, **Arrival**, **Amp**, **Mec**, **Coda** |
| Description: | The *auth* field specifies the source of the information. This may be a network identifier (FDSN or otherwise), or any other string indicating the source of the information in the record. |
| Datatype: | ***char(15)*** |
| NULL Value: | ***No*** |
| Range: | Any free-format string up to 15 characters |

| | |
|---|---|
| Name: | *aziinter* |
| Relations: | **Origin** |
| Description: | Azimuth of intermediate principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq aziinter \leq 360.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *azilarge* |
| Relations: | **Origin** |
| Description: | Azimuth of largest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq azilarge \leq 360.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *azimuth* |
| Relations: | **Arrival** |
| Description: | Observed azimuth. This is the estimated station-to-event azimuth measured clockwise from north. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq azimuth \leq 360.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *azismall* |
| Relations: | **Origin** |
| Description: | Azimuth of smallest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq azismall \leq 360.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *azres* |
| Relations: | **AssocArO** |
| Description: | Azimuth residual. This is the difference between the measured station-to-event azimuth for an arrival and the true azimuth. The "true" azimuth is the bearing to the inferred event origin. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $-180.0 \leq azres \leq 180.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *channel* |
| Relations: | **Arrival**, **Amp**, **Coda** |
| Description: | Channel name. |
| Datatype: | ***char(3)*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid channel name, which may be network-specific. Networks using SEED channel names should use the same value for *channel* and *seedchan* |

| | |
|---|---|
| Name: | *channelsrc* |
| Relations: | **Arrival**, **Amp**, **Coda** |
| Description: | Domain for channel. This specifies what naming convention is used for the channel name (i.e. SEED, USGS, etc.). |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 8 characters long |

| | |
|---|---|
| Name: | *cmodelid* |
| Relations: | **Origin** |
| Description: | Crustal model identifier. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *cmodelid* > 0 |

| | |
|---|---|
| Name: | *codatype* |
| Relations: | **Coda** |
| Description: | Coda type. |
| Datatype: | ***char(3)*** |
| NULL Value: | ***Yes*** |
| Range: | *codatype* ∈ {P \| S}<br>P = USGS P-wave coda<br>S = UCB S-wave coda |

| | |
|---|---|
| Name: | *coid* |
| Relations: | **Coda**, **AssocCoM**, **AssocCoO** |
| Description: | Coda identification. Each coda is assigned a unique positive integer which identifies it in the database. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *coid* > 0 |

| | |
|---|---|
| Name: | *commid* |
| Relations: | **Event**, **Origin**, **Netmag**, **Arrival**, **AssocArO**, **Amp**, **AssocAmO**, **AssocAmM**, **Remark**, **Mec**, **Coda**, **AssocCoM**, **AssocCoO** |
| Description: | Comment identification. This is a key used to point to free-form comments entered in the **Remark** relation. These comments store additional information about a tuple in another relation. Within the **Remark** relation, there may be many tuples with the same *commid* and different *lineno*, but the same *commid* will appear in only one other tuple among the rest of the relations in the database. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** (***No*** in the **Remark** relation) |
| Range: | *commid* > 0 |

| | |
|---|---|
| Name: | *datetime* |
| Relations: | **Origin**, **Arrival**, **Amp**, **Mec** |
| Description: | The date of associated with information in the record, in true epoch format. |
| Datatype: | ***float*** |
| NULL Value: | **No** |
| Range: | Any valid float. *datetime* = 0.0 on **January 01, 1970** at **00:00:00.0000** |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *datumhor* |
| Relations: | **Origin** |
| Description: | Datum type for horizontal (latitude and longitude) coordinates. |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | *datumhor* ∈ { NAD27 \| WGS84} |

| | |
|---|---|
| Name: | *datumver* |
| Relations: | **Origin** |
| Description: | Datum type for depth or elevation. |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | *datumver* ∈ { NAD27 \| WGS84 \| AVERAGE}<br>AVERAGE = average of stations |

| | |
|---|---|
| Name: | *delaz* |
| Relations: | **Arrival** |
| Description: | Delta azimuth. This attribute gives the standard deviation of the azimuth of a signal. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *delaz* > 0.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *delinc* |
| Relations: | **Arrival** |
| Description: | Error in angle. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *delinc* ≥ 0.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *delslo* |
| Relations: | **Arrival** |
| Description: | Delta slowness. This attribute gives the standard deviation of the slowness of a signal. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *delslo* > 0.0 |
| Units: | ***Seconds/Kilometers*** |

| | |
|---|---|
| Name: | *delta* |
| Relations: | **AssocArO**, **AssocAmO** |
| Description: | Source-receiver distance. This attribute is the arc length, over the earth's surface, of the path the seismic phase follows from source to receiver. The location of the origin is specified in the origin record referenced by the attribute *orid*. The attribute *arid* points to the record in the **Arrival** relation that identifies the receiver. The value of the attribute can exceed 180 degrees, it can even exceed 360 degrees, the geographic distance between source and receiver is *delta* mod(180). |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *delta* ≥ 0.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *deltim* |
| Relations: | **Arrival** |
| Description: | Delta time. This attribute gives the standard deviation of a detection time. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *deltim* ≥ 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *depth* |
| Relations: | **Origin** |
| Description: | Source depth. This attribute gives the depth of the event origin. The reference depth of 0 is usually defined as the average elevation of stations used to determine the origin. Positive values are down, negative values are up from the reference point. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -10.0 ≤ *depth* ≤ 1000.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *dipinter* |
| Relations: | **Origin** |
| Description: | Dip of intermediate principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -90.0 ≤ *dipinter* ≤ 90.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *diplarge* |
| Relations: | **Origin** |
| Description: | Dip of largest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -90.0 ≤ *diplarge* ≤ 90.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *dipsmall* |
| Relations: | **Origin** |
| Description: | Dip of smallest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -90.0 ≤ *dipsmall* ≤ 90.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *dip1* |
| Relations: | **Mec** |
| Description: | Dip 1 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | -90 ≤ *dip1* ≤ 90 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *dip2* |
| Relations: | **Mec** |
| Description: | Dip 2 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | -90 ≤ *dip2* ≤ 90 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *distance* |
| Relations: | **Origin**, **Netmag** |
| Description: | Distance to nearest station. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *distance* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *durtype* |
| Relations: | **Amp**, **Coda** |
| Description: | Duration type. |
| Datatype: | ***char(3)*** |
| NULL Value: | ***Yes*** |
| Range: | *durtype* ∈ {S}<br>S = S-wave |

| | |
|---|---|
| Name: | *eigenn* |
| Relations: | **Mec** |
| Description: | Eigen null. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |

| | |
|---|---|
| Name: | *eigenp* |
| Relations: | **Mec** |
| Description: | Eigen pressure. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |

| | |
|---|---|
| Name: | *eigent* |
| Relations: | **Mec** |
| Description: | Eigen tension. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |

| | |
|---|---|
| Name: | *ema* |
| Relations: | **Arrival** |
| Description: | Emergence angle. This attribute is the emergence angle of an arrival, as observed at a three-component station or array. The value increases from the vertical direction towards the horizontal. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq ema \leq 90.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *emares* |
| Relations: | **AssocArO** |
| Description: | Emergence angle residual. This attribute is the difference between an observed emergence angle and the theorical prediction for the same phase, assuming an event location as specified by the accompanying *orid*. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -90.0 ≤ *emares* ≤ 90.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *eramp* |
| Relations: | **Amp**, **Coda** |
| Description: | Error in amplitude. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *eramp* ≥ 0.0 |

| | |
|---|---|
| Name: | *erhor* |
| Relations: | **Origin** |
| Description: | Horizontal error. This attribute denotes the location uncertainty that accompanies the location. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *erhor* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *erlat* |
| Relations: | **Origin** |
| Description: | Error in latitude. This is the maximum error of the latitude estimate. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *erlat* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *erlon* |
| Relations: | **Origin** |
| Description: | Error in longitude. This is the maximum error of the longitude estimate. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *erlon* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *erscalar* |
| Relations: | ***Mec*** |
| Description: | Error in scalar moment. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *erscalar* ≥ 0.0 |
| Units: | ***Dynes centimeters*** |

| | |
|---|---|
| Name: | *etype* |
| Relations: | **Event** |
| Description: | This attribute is used to identify the type of seismic event, when known. |
| Datatype: | ***char(7)*** |
| NULL Value: | ***Yes*** |
| Range: | *etype* ∈ {le \| re \| ts \| qb \| nt \| uk \| sn}<br>le = local event<br>re = regional event<br>ts = teleseism<br>qb = quarry blast<br>nt = nuclear test<br>uk = unknown event<br>sn = sonic blast |

| | |
|---|---|
| Name: | *evid* |
| Relations: | **Event**, **Significant_Event**, **Origin**, **Netmag**, **Arrival**, **Amp**, **Mec**, **Coda** |
| Description: | Event identifier. Each event is assigned a unique integer which identifies it in a database. It is possible for several records in the **Origin** relation to have the same *evid*. This indicates there are several opinions about the location of the event. The *evid* does not have to be identical to the local network's event identifier, which is stored in the **Origin** table. This allows to merge data for a single event from multiple sources without insisting that database *evid* is identical to the source's local event identifier. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *evid* > 0 |

| | |
|---|---|
| Name: | *evname* |
| Relations: | **Significant_Event** |
| Description: | Event name. This is a descriptive name for major events. |
| Datatype: | ***char(80)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 80 characters |

| | |
|---|---|
| Name: | *exp* |
| Relations: | **Mec** |
| Description: | Exponent of moment tensor elements. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq exp \leq 32$ |

| | |
|---|---|
| Name: | *fdepth* |
| Relations: | **Origin** |
| Description: | This attribute indicates if the depth was fixed or not to compute the solution. |
| Datatype: | ***char(1)*** |
| NULL Value: | ***Yes*** |
| Range: | *fdepth* ∈ {y \| n}<br>y = fixed depth<br>n = free depth |

| | |
|---|---|
| Name: | *fepi* |
| Relations: | **Origin** |
| Description: | This attribute indicates if the epicenter was fixed or not to compute the solution. |
| Datatype: | ***char(1)*** |
| NULL Value: | ***Yes*** |
| Range: | *fepi* ∈ {y \| n}<br>y = fixed epicenter<br>n = free epicenter |

| | |
|---|---|
| Name: | *flagamp* |
| Relations: | **Amp** |
| Description: | This attribute is a flag to indicate whether amplitude is over P packet, S packet, entire waveform, etc. |
| Datatype: | ***char(4)*** |
| NULL Value: | ***Yes*** |
| Range: | *flagamp* ∈ {P \| S \| ALL}<br>P = P packet<br>S = S packet<br>ALL = entire waveform |

| | |
|---|---|
| Name: | *fm* |
| Relations: | **Arrival** |
| Description: | First motion. This is a two-character indication of first motion. The first character describes first motion seen on short-period channels and the second holds for long-period instruments.<br>Compression (dilation) on a short-period sensor is denoted by c(d) and compression (dilation) on a long-period sensor is denoted by u(r). Empty character positions will be indicated by dots (e.g., ".r"). |
| Datatype: | ***char(2)*** |
| NULL Value: | ***Yes*** |
| Range: | All two-letter permutations of (c \| d \| .} ∪ {u \| r \| .} |

| | |
|---|---|
| Name: | *ftime* |
| Relations: | **Origin** |
| Description: | This attribute indicates if the origin time was fixed or not to compute the solution. |
| Datatype: | ***char(1)*** |
| NULL Value: | ***Yes*** |
| Range: | *ftime* ∈ {y \| n}<br>y = fixed origin time<br>n = free origin time |

| | |
|---|---|
| Name: | *gap* |
| Relations: | **Origin**, **Netmag** |
| Description: | Azimuthal gap. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq gap \leq 360.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *importance* |
| Relations: | **AssocArO**, **AssocAmO** |
| Description: | This attribute denotes the importance of a phase or an amplitude for a solution. 0.0 means no importance, 1.0 implies extremely important. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 < importance \leq 1.0$ |

| | |
|---|---|
| Name: | *iphase* |
| Relations: | **Arrival**, **AssocArO**, **Coda** |
| Description: | Reported phase. This eight-character field holds the name initially given to a seismic phase. Standard seismological labels for the types of signals (or phases) are used (e.g., P, PKP, PcP, pP). Both upper and lower case letters are available and should be used when appropriate, for example, pP or PcP. |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 8 characters |

| | |
|---|---|
| Name: | *lat* |
| Relations: | **Origin** |
| Description: | Latitude. This attribute is the geographic latitude. Locations north of the equator have positive latitudes. |
| Datatype: | ***float*** |
| NULL Value: | ***No*** |
| Range: | -90.0 ≤ *lat* ≤ 90.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *lddate* |
| Relations: | **Event**, **Significant_Event**, **Origin**, **Netmag**, **Arrival**, **AssocArO**, **Amp**, **AssocAmO**, **AssocAmM**, **Remark**, **Lastid**, **Mec**, **Coda**, **AssocCoM**, **AssocCoO** |
| Description: | Load date. Date and time that the record was created or last modified, in Oracle ***date*** datatype. |
| Datatype: | ***date*** |
| NULL Value: | ***No*** |
| Range: | Any valid date between **January 01, 4712 BC** and **January 01, 4712 AD** |
| Units: | ***YYYY/MM/DD HH24:MI:SS*** |

| | |
|---|---|
| Name: | *lineno* |
| Relations: | **Remark** |
| Description: | Comment line number. This integer attribute is assigned as a sequence number for multiple line comments. The combination of *commid* and *lineno* is unique. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *lineno* > 0 |

| | |
|---|---|
| Name: | *locevid* |
| Relations: | **Origin** |
| Description: | Local event identifier. This attribute describes the original event identifier of the event. In combination with *auth* and *subsource*, it can be used to uniquely identify the original event identifier from that source. For example, waveforms from the original source may be associated with this identifier. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid integer |

| | |
|---|---|
| Name: | *lon* |
| Relations: | **Origin** |
| Description: | Longitude. This attribute is the geographic longitude. Longitudes are measured positive east of the Greenwich meridian. |
| Datatype: | ***float*** |
| NULL Value: | ***No*** |
| Range: | $-180.0 \leq lon \leq 180.0$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *mag* |
| Relations: | **AssocAmM** |
| Description: | Magnitude for this station reading. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $-10.0 < mag < 10.0$ |

| | |
|---|---|
| Name: | *magalgo* |
| Relations: | **Netmag** |
| Description: | This attribute represents the algorithm used to compute the magnitude. |
| Datatype: | ***char(15)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 15 characters |

| | |
|---|---|
| Name: | *magcorr* |
| Relations: | **AssocAmM** |
| Description: | Magnitude correction for this station. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $-10.0 \leq magcorr \leq 10.0$ |
| Units: | ***Magnitude units*** |

| | |
|---|---|
| Name: | *magid* |
| Relations: | **Netmag**, **AssocAmM**, **Mec**, **AssocCoM** |
| Description: | Magnitude identifier. This key is used to identify a magnitude in the **Netmag** relation. It is required for every magnitude. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | $magid > 0$ |

| | |
|---|---|
| Name: | *maginter* |
| Relations: | **Origin** |
| Description: | Magnitude of intermediate principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *maginter* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *maglarge* |
| Relations: | **Origin** |
| Description: | Magnitude of largest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *maglarge* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *magnitude* |
| Relations: | **Netmag** |
| Description: | Magnitude. This gives the magnitude value of the type indicated by *magtype*. |
| Datatype: | ***float*** |
| NULL Value: | ***No*** |
| Range: | -10.0 < *magnitude* < 10.0 |

| | |
|---|---|
| Name: | *magres* |
| Relations: | **AssocAmM** |
| Description: | Magnitude residual. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *magres* ≥ 0.0 |

| | |
|---|---|
| Name: | *magsmall* |
| Relations: | **Origin** |
| Description: | Magnitude of smallest principal error. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *magsmall* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *magtype* |
| Relations: | **Netmag** |
| Description: | Magnitude type. This character string is used to specify the type of the magnitude measure. |
| Datatype: | ***char(6)*** |
| NULL Value: | ***No*** |
| Range: | *magtype* ∈ {a \| b \| e \| l \| l1 \| l2 \| lg \| c \| s \| w \| z \| B \| un \| d \| h \| n}<br>a = primary amplitude magnitude (Jerry Eaton's XMAG)<br>b = body-wave magnitude<br>e = energy magnitude<br>l = local magnitude<br>l1 = traditional UCB local magnitude<br>l2 = network UCB local magnitude<br>lg = Lg magnitude<br>c = primary coda magnitude<br>s = Ms – surface wave magnitude<br>w = moment magnitude<br>z = low gain (Z component) magnitude of Hirshorn and Lindh (1989)<br>B = magnitude estimated from 14-kg Benioff's<br>un = unknown magnitude type<br>d = duration magnitude<br>h = helicorder magnitude (CIT, short-period Benioff)<br>n = no magnitude |

| | |
|---|---|
| Name: | *mecid* |
| Relations: | **Mec** |
| Description: | Mechanism identification. Each mechanism is assigned a unique positive integer which identifies it in the database. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *mecid* > 0 |

| | |
|---|---|
| Name: | *mecalgo* |
| Relations: | **Mec** |
| Description: | This attribute represents the algorithm used to compute the mechanism. |
| Datatype: | ***char(15)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 15 characters |

| | |
|---|---|
| Name: | *mechtype* |
| Relations: | **Mec** |
| Description: | Mechanism/moment type. |
| Datatype: | ***char(2)*** |
| NULL Value: | ***Yes*** |
| Range: | *mechtype* ∈ {C \| C0 \| F \| F0 \| H \| N \| S \| S0}<br>C = complete-waveform inversion<br>C0 = Dreger's waveform inversion<br>F = first motion mechanism<br>F0 = fpfit<br>H = half duration of S-wave<br>N = near-field inversion<br>S = surface-wave inversion<br>S0 = Romanowicz & Pasyanos surface-wave inversion |

| | |
|---|---|
| Name: | *mmi* |
| Relations: | **Significant_Event** |
| Description: | Reported or inferred Mercali intensity for event. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $1 \leq mmi \leq 12$ |

| | |
|---|---|
| Name: | *mxx, myy, mzz, mxy, mxz, myz* |
| Relations: | **Mec** |
| Description: | Moment tensor elements. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -10.0 < *mxx, myy, mzz, mxy, mxz, myz* < 10.0 |
| Units: | ***Dynes centimeters*** |

| | |
|---|---|
| Name: | *nbp* |
| Relations: | **Origin** |
| Description: | Number of P first motions. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $nbp \geq 0$ |

| | |
|---|---|
| Name: | *nbs* |
| Relations: | **Origin** |
| Description: | Number of S phase readings. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $nbs \geq 0$ |

| | |
|---|---|
| Name: | *ndef* |
| Relations: | **Origin** |
| Description: | Number of time-defining phases. This attribute is the number of arrivals used to locate an event. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $ndef \geq 0$ |

| | |
|---|---|
| Name: | *net* |
| Relations: | **Arrival**, **Amp**, **Coda** |
| Description: | Unique network identifier. This character string is the name of a seismic network. |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 8 characters. If available, use the two character FDSN code for the network |

| | |
|---|---|
| Name: | *nfelt* |
| Relations: | **Significant_Event** |
| Description: | Number of felt reports. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $nfelt > 0$ |

| | |
|---|---|
| Name: | *nsample* |
| Relations: | **Coda** |
| Description: | Number of coda sample windows. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *nsample* > 0 |

| | |
|---|---|
| Name: | *nsta* |
| Relations: | **Netmag**, **Mec** |
| Description: | Number of stations. This quantity is the number of stations used to compute the magnitude or the mechanism of the event. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *nsta* > 0 |

| | |
|---|---|
| Name: | *orid* |
| Relations: | **Origin**, **Netmag**, **AssocArO**, **AssocAmO**, **AssocCoO** |
| Description: | Origin identification. Each origin is assigned a unique positive integer which identifies it in the database. The *orid* is used to identify one of the many hypotheses of the actual location of the event. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *orid* > 0 |

| | |
|---|---|
| Name: | *oridin* |
| Relations: | **Mec** |
| Description: | Input origin. This attribute holds the origin identifier, *orid*, that points to the origin from which the mechanism was computed. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *oridin > 0* |

| | |
|---|---|
| Name: | *oridout* |
| Relations: | **Mec** |
| Description: | Output origin. This attribute holds the origin identifier, *orid*, that points to the origin computed from the mechanism. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *oridout > 0* |

| | |
|---|---|
| Name: | *per* |
| Relations: | **Amp** |
| Description: | Signal period. This attribute is the period of the signal described by the *amplitude* record. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *per* > 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *pga* |
| Relations: | **Significant_Event** |
| Description: | Peak ground acceleration recorded by any station for the event.Represented in g, where 1.0 = 1g. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *pga* > 0.0 |
| Units: | ***g*** |

| | |
|---|---|
| Name: | *plungen* |
| Relations: | **Mec** |
| Description: | Plunge null. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq plungen \leq 90$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *plungep* |
| Relations: | **Mec** |
| Description: | Plunge pressure. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq plungep \leq 90$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *plunget* |
| Relations: | **Mec** |
| Description: | Plunge tension. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq plunget \leq 90$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *prefmag* |
| Relations: | **Event**, **Origin** |
| Description: | Preferred magnitude. This attribute holds the magnitude identifier, *magid*, that points to the preferred magnitude for a seismic event or an origin. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $prefmag > 0$ |

| | |
|---|---|
| Name: | *prefmec* |
| Relations: | **Event**, **Origin** |
| Description: | Preferred mechanism. This attribute holds the mechanism identifier, *mecid*, that points to the preferred mechanism for a seismic event or an origin. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $prefmec > 0$ |

| | |
|---|---|
| Name: | *prefor* |
| Relations: | **Event** |
| Description: | Preferred origin. This attribute holds the origin identifier, *orid*, that points to the preferred origin for a seismic event. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** (only if no origin exists for the event) |
| Range: | *prefor* > 0 |

| | |
|---|---|
| Name: | *pclvd* |
| Relations: | **Mec** |
| Description: | Percent compensated linear vector dipole. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq pclvd \leq 100$ |

| | |
|---|---|
| Name: | *pdc* |
| Relations: | **Mec** |
| Description: | Percent double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq pdc \leq 100$ |

| | |
|---|---|
| Name: | *piso* |
| Relations: | **Mec** |
| Description: | Percent isotropic. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq piso \leq 100$ |

| | |
|---|---|
| Name: | *pvr* |
| Relations: | **Mec** |
| Description: | Percent variance reduction. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq pvr \leq 100.0$ |

| | |
|---|---|
| Name: | *qfix* |
| Relations: | **Coda** |
| Description: | Fixed coda decay constant. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid float |

| | |
|---|---|
| Name: | *qfree* |
| Relations: | **Coda** |
| Description: | Free decay. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid float |

| | |
|---|---|
| Name: | *qual* |
| Relations: | **Arrival** |
| Description: | Onset quality. This single-character flag is used to denote the sharpness of the onset of a seismic phase. |
| Datatype: | ***char(1)*** |
| Null Value: | ***Yes*** |
| Range: | *qual* ∈ {i \| e \| w}<br>i = impulsive<br>e = emergent<br>w = weak |

| | |
|---|---|
| Name: | *quality* |
| Relations: | **Origin**, **Arrival**, **Mec** |
| Description: | This attribute denotes the quality of an origin, an arrival, or a mechanism.<br>1.0 = good, 0.0 = bad. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq quality \leq 1.0$ |

| | |
|---|---|
| Name: | *rake1* |
| Relations: | **Mec** |
| Description: | Rake 1 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | -180 ≤ *rake1* ≤ 180 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *rake2* |
| Relations: | **Mec** |
| Description: | Rake 2 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | -180 ≤ *rake2* ≤ 180 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *remark* |
| Relations: | **Remark** |
| Description: | Descriptive text. This single line of text is an arbitrary comment about a record in the database. The comment is linked to its "parent" relation only by forward reference from *commid* in the tuple of the relation of interest. |
| Datatype: | ***char(80)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 80 characters long |

| | |
|---|---|
| Name: | *remarks* |
| Relations: | **Significant_Event** |
| Description: | This attribute is used to give some information about the event, such as felt or damage. |
| Datatype: | ***char(2)*** |
| NULL Value: | ***Yes*** |
| Range: | *remarks* ∈ {f \| d}<br>f = felt<br>d = damage |

| | |
|---|---|
| Name: | *rms* |
| Relations: | **Coda** |
| Description: | Residual of mean squares for coda fit. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *rms* ≥ 0.0 |

| | |
|---|---|
| Name: | *scalar* |
| Relations: | ***Mec*** |
| Description: | Scalar moment. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -10.0 < *scalar* < 10.0 |
| Units: | ***Dynes centimeters*** |

| | |
|---|---|
| Name: | *scorr* |
| Relations: | **AssocArO** |
| Description: | Station correction. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid float |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *sdelay* |
| Relations: | **AssocArO** |
| Description: | Station delay. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | Any valid float |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *sdepth* |
| Relations: | **Origin** |
| Description: | Depth error. This is the maximum error of the depth estimate. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *sdepth* ≥ 0.0 |
| Units: | ***Kilometers*** |

| | |
|---|---|
| Name: | *sdobs* |
| Relations: | **Origin** |
| Description: | Standard error of one observation. This attribute is derived from the discrepancies in the arrival times of the phases used to locate an event. It is defined as the square root of the sum of the squares of the time residuals, divided by the number of degrees of freedom. The latter is the number of defining observations (*ndef* in **Origin**) minus the dimension of the system solved (4 if depth is allowed to be a free variable, 3 if depth is constrained). |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *sdobs* > 0.0 |

| | |
|---|---|
| Name: | *seaz* |
| Relations: | **AssocArO**, **AssocAmO** |
| Description: | Station to event azimuth. This attribute is calculated from the station and event locations. It is measured clockwise from North. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq seaz \leq 360.0$ |
| Units: | ***Degrees*** |

Name: *seedchan*

Relations: **Arrival**, **Amp**, **Coda**

Description: SEED channel name. The first character denotes the band code, the second holds for the instrument code and the third one represents the component code.

Datatype: ***char(3)***

NULL Value: ***Yes***

Range: *seedchan* ∈ {E | S | H | B | M | L | V | U | R} ∪
{A | B | D | F | G | H | I | K | L | M | P | R | S | V | T | W} ∪
{Z | N | E | A | B | C | T | R | 1 | 2 | 3 | U | V | W}

Band code:

| | Sample rate (Hz) | Corner period (sec) |
|---|---|---|
| E = extremely short period | ≥ 80 | < 10 |
| S = short period | ≥ 10 to < 80 | < 10 |
| H = high broad band | ≥ 80 | ≥ 10 |
| B = broad band | ≥ 10 to < 80 | ≥ 10 |
| M = mid period | > 1 to < 10 | |
| L = long period | ~ 1 | |
| V = very long period | ~ 0.1 | |
| U = ultra long period | ~ 0.01 | |
| R = extremely long period | ~ 0.001 | |

Instrument code:

| | |
|---|---|
| A = tiltmeter | B = creepmeter |
| D = pressure | F = magnetometer |
| G = gravimeter/accelerometer | H = high gain seismometer |
| I = humidity | K = temperature |
| L = low gain seismometer | M = mass position seismometer |
| P = geophone | R = rainfall |
| S = linear strainmeter | V = volumetric strain |
| T = tide meter | W = wind |

Component code:

| | |
|---|---|
| N = north-south | E = east-west |
| T = transverse | R = radial |

Z = vertical
A, B, C = triaxial
1, 2, 3 = orthogonal components but non-traditional orientations
U, V, W = optional components

| | |
|---|---|
| Name: | *slores* |
| Relations: | **AssocArO** |
| Description: | Slowness residual. This attribute gives the difference between an observed slowness and a theorical prediction. The prediction is calculated for the related phase and event origin described in the record. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *slores* ≥ 0.0 |
| Units: | ***Seconds/kilometer*** |

| | |
|---|---|
| Name: | *slow* |
| Relations: | **Arrival** |
| Description: | Observed slowness. This is the observed slowness of a wave as it sweeps across an array. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *slow* ≥ 0.0 |
| Units: | ***Seconds/kilometer*** |

| | |
|---|---|
| Name: | *snr* |
| Relations: | **Arrival**, **Amp** |
| Description: | Signal-to-noise ratio. This is an estimate of the signal relative to that of the noise immediately preceding it. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *snr* > 0.0 |

| | |
|---|---|
| Name: | *srcduration* |
| Relations: | **Mec** |
| Description: | Source half-duration. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | 0.0 < *srcduration* < 100.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *sta* |
| Relations: | **Arrival**, **Amp**, **Coda** |
| Description: | Station code. This is the common code-name of a seismic observatory. Generally only three or four characters are used. |
| Datatype: | ***char(6)*** |
| NULL Value: | ***No*** |
| Range: | Any free-format string up to 6 characters |

| | |
|---|---|
| Name | *stime* |
| Relations: | **Origin** |
| Description: | Origin time error. This attribute denotes the time uncertainty that accompanies the location. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *stime* ≥ 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *striken* |
| Relations: | **Mec** |
| Description: | Strike null. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq striken \leq 360$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *strikep* |
| Relations: | **Mec** |
| Description: | Strike pressure. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq strikep \leq 360$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *striket* |
| Relations: | **Mec** |
| Description: | Strike tension. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq striket \leq 360$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *strike1* |
| Relations: | **Mec** |
| Description: | Strike 1 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq strike1 \leq 360$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *strike2* |
| Relations: | **Mec** |
| Description: | Strike 2 of best double couple. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | $0 \leq strike2 \leq 360$ |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *stx*, *sty*, *stz*, *sxx*, *sxy*, *sxz*, *syy*, *syz*, *stt*, *szz* |
| Relations: | **Origin** |
| Description: | Elements of the covariance matrix for the location defined by *orid*. The covariance matrix is symmetric (and positive definite) so that sxy = syx, etc., (x, y, z, t) refer to latitude, longitude, depth and origin time, respectively. These attributes (together with *sdobs*, *ndef* and *dattype*) provide all the information necessary to construct the K-dimensional (K=2, 3, 4) confidence ellipse or ellipsoids at any confidence limit desired. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *sxx*, *syy*, *szz*, *stt* > 0.0 |
| Units: | *sxx*, *syy*, *szz*, *sxy*, *sxz*, *syz* – ***Kilometers squared***<br>*stt* – ***Seconds squared***<br>*stx*, *sty*, *stz* – ***Kilometers/second*** |

| | |
|---|---|
| Name: | *subsource* |
| Relations: | **Event**, **Origin**, **Netmag**, **Arrival**, **Amp**, **Mec**, **Coda** |
| Description: | A second identifier to specify a unique source within the domain specified by *auth*. |
| Datatype: | ***char(8)*** |
| NULL Value: | ***Yes*** |
| Range: | Any free-format string up to 8 characters |

| | |
|---|---|
| Name: | *tau* |
| Relations: | **Amp**, **Coda** |
| Description: | Coda duration (F-P time) |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *tau* > 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *tfd* |
| Relations: | ***Mec*** |
| Description: | Time function duration. |
| Datatype: | ***float*** |
| Range: | *tfd* > 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *tft* |
| Relations: | ***Mec*** |
| Description: | Time function type. |
| Datatype: | ***char(8)*** |
| Range: | Any free-format string up to 8 characters |

| | |
|---|---|
| Name: | *time1, time2, time3, time4, time5, time6* |
| Relations: | **Coda** |
| Description: | Time for pairs. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *time1, time2, time3, time4, time5, time6* > 0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *timeres* |
| Relations: | **AssocArO** |
| Description: | Time residual. This attribute is a travel time residual, measured in seconds. The residual is found by taking the observed arrival time (saved in the **Arrival** relation) of a seismic phase and substracting the expected arrival time. The expected arrival time is calculated by a formula based on earth velocity model, an event location and origin time, the distance to the station, and the particular seismic phase. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *timeres* ≥ 0.0 |
| Units: | ***Seconds*** |

| | |
|---|---|
| Name: | *totalamp* |
| Relations: | **Event** |
| Description: | Total number of amplitudes in the database for the specified *evid*. Not all amplitudes have to associated with all (or any) magnitudes or origins. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *totalamp* ≥ 0 |

| | |
|---|---|
| Name: | *totalarr* |
| Relations: | **Event** |
| Description: | Total number of arrivals in the database for the specified *evid*. Not all arrivals have to associated with all (or any) origins. |
| Datatype: | ***int*** |
| NULL Value: | ***No*** |
| Range: | *totalarr* ≥ 0 |

| | |
|---|---|
| Name: | *type* |
| Relations: | **Origin** |
| Description: | Type of location. |
| Datatype: | ***char(2)*** |
| NULL Value: | ***Yes*** |
| Range: | *type* ∈ {H \| C \| A}<br>H = hypocenter<br>C = centroid<br>A = amplitude |

| | |
|---|---|
| Name: | *uncertainty* |
| Relations: | **Netmag** |
| Description: | Magnitude uncertainty. This is the standard deviation of the accompanying magnitude measurement. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | *uncertainty* > 0.0 |

| | |
|---|---|
| Name: | *undip1* |
| Relations: | **Mec** |
| Description: | Uncertainty in dip 1. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *undip1* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *undip2* |
| Relations: | **Mec** |
| Description: | Uncertainty in dip 2. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *undip2* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *units* |
| Relations: | **Amp**, **Coda** |
| Description: | Units of amplitude. |
| Datatype: | ***char(4)*** |
| NULL Value: | ***No*** |
| Range: | *units* ∈ {c \| s \| mm \| cm \| m \| ms \| mss \| cms \| cmss \| mms \| mmss \| mc \| nm}<br>c = counts<br>s = seconds<br>mm = millimeters<br>cm = centimeters<br>m = meters<br>ms = meters per second<br>mss = meters per second per second<br>cms = centimeters per second<br>cmss = centimeters per second per second<br>mms = millimeters per second<br>mmss = millimeters per second per second<br>mc = microns<br>nm = nanometers |

| | |
|---|---|
| Name: | *unmxx, unmyy, unmzz, unmxy, unmxz, unmyz* |
| Relations: | ***Mec*** |
| Description: | Uncertainties in moment tensor elements. |
| Datatype: | ***float*** |
| Range: | -10.0 < *unmxx, unmyy, unmzz, unmxy, unmxz, unmyz* < 10.0 |
| Units: | ***Dynes centimeters*** |

| | |
|---|---|
| Name: | *unrake1* |
| Relations: | **Mec** |
| Description: | Uncertainty in rake 1. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *unrake1* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *unrake2* |
| Relations: | **Mec** |
| Description: | Uncertainty in rake 2. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *unrake2* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *unstrike1* |
| Relations: | **Mec** |
| Description: | Uncertainty in strike 1. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *unstrike1* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *unstrike2* |
| Relations: | **Mec** |
| Description: | Uncertainty in strike 2. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | -180.0 ≤ *unstrike2* ≤ 180.0 |
| Units: | ***Degrees*** |

| | |
|---|---|
| Name: | *vmodelid* |
| Relations: | **Origin** |
| Description: | Velocity model identification. |
| Datatype: | ***int*** |
| NULL Value: | ***Yes*** |
| Range: | *vmodelid* > 0 |

| | |
|---|---|
| Name: | *weight* |
| Relations: | **AssocAmM** |
| Description: | Weight code. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | 0.0 < *weight* ≤ 1.0<br>0.0 = no weight (not used) |

| | |
|---|---|
| Name: | *wgt* |
| Relations: | **AssocArO** |
| Description: | Location weight. This attribute gives the final weight assigned to the allied arrival by the location program. It is used primarily for location programs that adaptively weight data by their residuals. |
| Datatype: | ***float*** |
| NULL Value: | ***Yes*** |
| Range: | $0.0 \leq wgt < 1.0$ |

***5.- Primary keys :***

Oracle's sequences are used now to generate primary keys values instead of the **Lastid** table. A sequence is a database object from which multiple users may generate unique integers.

Values for a given sequence are automatically generated by special Oracle routines and, consequently, sequences avoid the performance bottleneck which results from implementation of sequences at the application level. For example, one common application-level implementation is to force each transaction to lock a sequence number table, increment the sequence, and then release the table. Under this implementation, only one sequence number may be generated at a time. In contrast, Oracle sequences permit the simultaneous generation of multiple sequence numbers while guaranteeing that every sequence number is unique.

When a sequence number is generated, the sequence is incremented, independent of the transaction committing or rolling back. If two users concurrently increment the same sequence, the sequence numbers each user acquires may have gaps because sequence numbers are being generated by the other user. One user can never acquire the sequence number generated by another user. Once a sequence value is generated by one user, that user can continue to access that value regardless of whether the sequence is incremented by another user.

Because sequence numbers are generated independently of tables, the same sequence can be used for one or for multiple tables. It is possible that individual sequence numbers will appear to be skipped, because they were generated and used in a transaction that ultimately rolled back. Additionally, a single user may not realize that other users are drawing from the same sequence.

For more information, see **Oracle8 Server SQL Reference Release 8.0**

***6.- Time issues :***

The following *PL/SQL* script ***date.sql*** is an implementation of the different functions to make the conversion between :

- String format time.
- Nominal epoch time.
- True epoch time (Atomic time TAI).

This script consists of six functions to convert between the different times, a procedure to populate the table needed for leap seconds, and an example. The origin time chosen to compute nominal and true epoch times is **January 01, 1970** at **00:00:00**.

To implement the concept of leap seconds, a table **leap_seconds** was needed. This table has the following structure :

| **Leap_seconds** | | Information about leap seconds |
|---|---|---|
| ***Attribute*** | ***Type*** | ***Description*** |
| s_nominal | int | Start nominal time |
| e_nominal | int | End nominal time |
| s_true | int | Start true time |
| e_true | int | End true time |
| ls_count | int | Leap seconds count |

Each row of the **leap_seconds** table represents an interval between two leap seconds.

In the script's example, there is 20 single leap seconds and a double leap second on **December 31, 1972** at **23:59:59**. In real life, there was only a single leap second at that time.

- *Function **string2nominal :***

This function converts a string format time to a nominal epoch time.

*Examples :*

***string2nominal*** ('1972/12/31 23:59:59') = 94694399
***string2nominal*** ('1972/12/31 23:59:60') = NULL
***string2nominal*** ('1972/12/31 23:59:61') = NULL
***string2nominal*** ('1973/01/01 00:00:00') = 94694400

- *Function* ***nominal2string :***

This function converts a nominal epoch time to a string format time.

*Examples :*

***nominal2string*** (0) = '1970/01/01 00:00:00'
***nominal2string*** (94694399) = '1972/12/31 23:59:59'
***nominal2string*** (94694400) = '1973/01/01 00:00:00'

- *Function* ***nominal2true :***

This function converts a nominal epoch time to a true epoch time.

*Examples :*

***nominal2true*** (0) = 0
***nominal2true*** (94694399) = 94694400
***nominal2true*** (94694400) = 94694403

- *Function* ***true2nominal :***

This function converts a true epoch time to a nominal epoch time.

*Examples :*

***true2nominal*** (94694400) = 94694399
***true2nominal*** (94694401) = NULL
***true2nominal*** (94694402) = NULL
***true2nominal*** (94694403) = 94694400

- *Function* ***string2true :***

This function converts a string format time to a true epoch time.

*Examples :*

***string2nominal*** ('1972/12/31 23:59:59') = 94694400
***string2nominal*** ('1972/12/31 23:59:60') = 94694401
***string2nominal*** ('1972/12/31 23:59:61') = 94694402
***string2nominal*** ('1973/01/01 00:00:00') = 94694403

- *Function **true2string** :*

This function converts a true epoch time to a string format time.

*Examples :*

***true2string*** (94694400) = '1972/12/31 23:59:59'
***true2string*** (94694401) = '1972/12/31 23:59:60'
***true2string*** (94694402) = '1972/12/31 23:59:61'
***true2string*** (94694403) = '1973/01/01 00:00:00'

```
 /*************************************************************************
 * date.sql ---> PL/SQL Script for conversion between :                   *
 *                                                                        *
 *          - String format time.                                         *
 *          - Nominal epoch time.                                         *
 *          - True epoch time.                                            *
 *                                                                        *
 * Stephane Zuzlewski @1998                                               *
 *                                                                        *
 * % sqlplus scott/tiger @date.sql                                        *
 *************************************************************************/

 /*************************************************************************
 * The table leap_seconds contains information about all the current     *
 * leap seconds.                                                          *
 *************************************************************************/

DROP TABLE leap_seconds;

CREATE TABLE leap_seconds (
s_nominal    INTEGER,     /* Start nominal epoch time */
e_nominal    INTEGER,     /* End nominal epoch time   */
s_true       INTEGER,     /* Start true epoch time    */
e_true       INTEGER,     /* End true epoch time      */
ls_count     INTEGER      /* Leap seconds count       */
);


 /*************************************************************************
 * The table leap_seconds_tmp is a temporary table needed to populate    *
 * the leap_seconds table.                                                *
 *************************************************************************/

DROP TABLE leap_seconds_tmp;

CREATE TABLE leap_seconds_tmp (
s_nominal    INTEGER,     /* Start nominal epoch time */
e_nominal    INTEGER,     /* End nominal epoch time   */
s_true       INTEGER,     /* Start true epoch time    */
e_true       INTEGER,     /* End true epoch time      */
ls_count     INTEGER      /* Leap seconds count       */
);
```

```
 /*************************************************************************
 * The table temp is a temporary table just here to test the different  *
 * functions.                                                            *
 *************************************************************************/

DROP TABLE temp;

CREATE TABLE temp (
string      VARCHAR(20),
nominal     INTEGER,
true        INTEGER
);



 /*************************************************************************
 * The function string2nominal converts a string format time to a       *
 * nominal epoch time.                                                   *
 *                                                                       *
 * Returns NULL if the time is a leap second.                            *
 *************************************************************************/

DROP FUNCTION string2nominal;

CREATE FUNCTION string2nominal (d VARCHAR) RETURN INTEGER IS
      origin      CONSTANT DATE := '1970/01/01 00:00:00';
      origin_days INTEGER;
      d_days      INTEGER;
      diff_days   INTEGER;
      nominal     INTEGER;
      d2          VARCHAR(20);
      d3          VARCHAR(20);
BEGIN
      d2          := REPLACE (d, ':60', ':59');
      d3          := REPLACE (d, ':61', ':59');

      IF    d != d2 THEN
            RETURN (NULL);
      ELSIF d != d3 THEN
            RETURN (NULL);
      ELSE
            origin_days := TO_CHAR (origin, 'J');
            d_days      := TO_CHAR (TO_DATE (d, 'YYYY/MM/DD HH24:MI:SS'),
                                    'J');

            diff_days   := d_days - origin_days;

            nominal     := (diff_days*86400) +
                               TO_CHAR (TO_DATE (d, 'YYYY/MM/DD HH24:MI:SS'),
                                        'SSSSS');

            RETURN (nominal);
      END IF;
END string2nominal;
/
```

```
 /*********************************************************************
 * The function nominal2string converts a nominal epoch time to a       *
 * string format time.                                                   *
 *********************************************************************/

DROP FUNCTION nominal2string;

CREATE FUNCTION nominal2string (n INTEGER) RETURN VARCHAR IS
      origin      CONSTANT DATE := '1970/01/01 00:00:00';
      d           VARCHAR(20);
BEGIN
      d           := TO_CHAR (TO_DATE (origin, 'YYYY/MM/DD HH24:MI:SS') +
                              (n/86400), 'YYYY/MM/DD HH24:MI:SS');

      RETURN (d);
END nominal2string;
/

 /*********************************************************************
 * The function nominal2true converts a nominal epoch time to a true    *
 * epoch time.                                                           *
 *********************************************************************/

DROP FUNCTION nominal2true;

CREATE FUNCTION nominal2true (n INTEGER) RETURN INTEGER IS
      true        INTEGER;
      lp_secs     INTEGER;
BEGIN
      SELECT ls_count INTO lp_secs FROM leap_seconds
      WHERE s_nominal <= n
      AND   e_nominal >= n;

      true        := n + lp_secs;

      RETURN (true);
END nominal2true;
/

 /*********************************************************************
 * The function true2nominal converts a true epoch time to a nominal    *
 * epoch time.                                                           *
 *                                                                       *
 * Returns NULL if the time is a leap second.                            *
 *********************************************************************/

DROP FUNCTION true2nominal;

CREATE FUNCTION true2nominal (t INTEGER) RETURN INTEGER IS
      nominal           INTEGER;
BEGIN
      nominal           := string2nominal (true2string (t));

      RETURN (nominal);
END true2nominal;
/
```

```
 /*********************************************************************
 * The function string2true converts a string format time to a true     *
 * epoch time.                                                          *
 *********************************************************************/

DROP FUNCTION string2true;

CREATE FUNCTION string2true (d VARCHAR) RETURN INTEGER IS
      true        INTEGER;
      leap        INTEGER;
      d2          VARCHAR(20);
      d3          VARCHAR(20);
      d4          VARCHAR(20);
BEGIN
      d2          := REPLACE (d, ':60', ':59');
      d3          := REPLACE (d, ':61', ':59');

      IF    d != d2 THEN
            d4    := d2;
            leap  := 1;
      ELSIF d != d3 THEN
            d4    := d3;
            leap  := 2;
      ELSE
            d4    := d;
            leap  := 0;
      END IF;


      true        := nominal2true (string2nominal (d4)) + leap;

      RETURN (true);
END string2true;
/
```

```
 /*********************************************************************
 * The function true2string converts a true epoch time to a string      *
 * format time.                                                         *
 *********************************************************************/

DROP FUNCTION true2string;

CREATE FUNCTION true2string (t INTEGER) RETURN VARCHAR IS
      d           VARCHAR(20);
      lp_secs1    INTEGER;
      lp_secs2    INTEGER;
      lp_secs3    INTEGER;
BEGIN

      BEGIN
      SELECT ls_count INTO lp_secs1 FROM leap_seconds
      WHERE s_true <= t
      AND   e_true >= t;
      EXCEPTION WHEN NO_DATA_FOUND THEN lp_secs1 := -1;
      END;

      IF    lp_secs1 >= 0     THEN
            d     := nominal2string (t - lp_secs1);
      ELSE
            BEGIN
            SELECT ls_count INTO lp_secs2 FROM leap_seconds
            WHERE s_true <= t-1
            AND   e_true >= t-1;
            EXCEPTION WHEN NO_DATA_FOUND THEN lp_secs2 := -1;
            END;

            IF    lp_secs2 >= 0     THEN
                  d     := nominal2string (t - lp_secs2 - 2);
                  d     := REPLACE (d, ':58', ':60');
            ELSE
                  BEGIN
                  SELECT ls_count INTO lp_secs3 FROM leap_seconds
                  WHERE s_true <= t-2
                  AND   e_true >= t-2;
                  END;

                  d     := nominal2string (t - lp_secs3 - 3);
                  d     := REPLACE (d, ':58', ':61');
            END IF;
      END IF;


      RETURN (d);
END true2string;
/
```

```
 /***********************************
 * Populating the leap_seconds table *
 ***********************************/

DECLARE

 /* Leap seconds in string format time */

l0      DATE := '0001/01/01 00:00:00'; /* Represents minus infinite */
l1      DATE := '1972/06/30 23:59:59';
l2      DATE := '1972/12/31 23:59:59'; /* Two leap seconds at this date */
l3      DATE := '1973/12/31 23:59:59';
l4      DATE := '1974/12/31 23:59:59';
l5      DATE := '1975/12/31 23:59:59';
l6      DATE := '1976/12/31 23:59:59';
l7      DATE := '1977/12/31 23:59:59';
l8      DATE := '1978/12/31 23:59:59';
l9      DATE := '1979/12/31 23:59:59';
l10     DATE := '1981/06/30 23:59:59';
l11     DATE := '1982/06/30 23:59:59';
l12     DATE := '1983/06/30 23:59:59';
l13     DATE := '1985/06/30 23:59:59';
l14     DATE := '1987/12/31 23:59:59';
l15     DATE := '1989/12/31 23:59:59';
l16     DATE := '1990/12/31 23:59:59';
l17     DATE := '1992/06/30 23:59:59';
l18     DATE := '1993/06/30 23:59:59';
l19     DATE := '1994/06/30 23:59:59';
l20     DATE := '1995/12/31 23:59:59';
l21     DATE := '1997/06/30 23:59:59';
l22     DATE := '3000/01/01 00:00:00'; /* Represents plus infinite */

 /* Leap seconds in nominal epoch time */
n0      INTEGER;
n1      INTEGER;
n2      INTEGER;
n3      INTEGER;
n4      INTEGER;
n5      INTEGER;
n6      INTEGER;
n7      INTEGER;
n8      INTEGER;
n9      INTEGER;
n10     INTEGER;
n11     INTEGER;
n12     INTEGER;
n13     INTEGER;
n14     INTEGER;
n15     INTEGER;
n16     INTEGER;
n17     INTEGER;
n18     INTEGER;
n19     INTEGER;
n20     INTEGER;
n21     INTEGER;
n22     INTEGER;
```

```
BEGIN
       /* Computing nominal epoch times */
        n0  := string2nominal (l0);
        n1  := string2nominal (l1);
        n2  := string2nominal (l2);
        n3  := string2nominal (l3);
        n4  := string2nominal (l4);
        n5  := string2nominal (l5);
        n6  := string2nominal (l6);
        n7  := string2nominal (l7);
        n8  := string2nominal (l8);
        n9  := string2nominal (l9);
        n10 := string2nominal (l10);
        n11 := string2nominal (l11);
        n12 := string2nominal (l12);
        n13 := string2nominal (l13);
        n14 := string2nominal (l14);
        n15 := string2nominal (l15);
        n16 := string2nominal (l16);
        n17 := string2nominal (l17);
        n18 := string2nominal (l18);
        n19 := string2nominal (l19);
        n20 := string2nominal (l20);
        n21 := string2nominal (l21);
        n22 := string2nominal (l22);

       /* Populating the leap_seconds table */
        INSERT INTO leap_seconds VALUES (n0, n1, NULL, NULL, 0);
        INSERT INTO leap_seconds VALUES (n1+1, n2, NULL, NULL, 1);
        INSERT INTO leap_seconds VALUES (n2+1, n3, NULL, NULL, 3);
        INSERT INTO leap_seconds VALUES (n3+1, n4, NULL, NULL, 4);
        INSERT INTO leap_seconds VALUES (n4+1, n5, NULL, NULL, 5);
        INSERT INTO leap_seconds VALUES (n5+1, n6, NULL, NULL, 6);
        INSERT INTO leap_seconds VALUES (n6+1, n7, NULL, NULL, 7);
        INSERT INTO leap_seconds VALUES (n7+1, n8, NULL, NULL, 8);
        INSERT INTO leap_seconds VALUES (n8+1, n9, NULL, NULL, 9);
        INSERT INTO leap_seconds VALUES (n9+1, n10, NULL, NULL, 10);
        INSERT INTO leap_seconds VALUES (n10+1, n11, NULL, NULL, 11);
        INSERT INTO leap_seconds VALUES (n11+1, n12, NULL, NULL, 12);
        INSERT INTO leap_seconds VALUES (n12+1, n13, NULL, NULL, 13);
        INSERT INTO leap_seconds VALUES (n13+1, n14, NULL, NULL, 14);
        INSERT INTO leap_seconds VALUES (n14+1, n15, NULL, NULL, 15);
        INSERT INTO leap_seconds VALUES (n15+1, n16, NULL, NULL, 16);
        INSERT INTO leap_seconds VALUES (n16+1, n17, NULL, NULL, 17);
        INSERT INTO leap_seconds VALUES (n17+1, n18, NULL, NULL, 18);
        INSERT INTO leap_seconds VALUES (n18+1, n19, NULL, NULL, 19);
        INSERT INTO leap_seconds VALUES (n19+1, n20, NULL, NULL, 20);
        INSERT INTO leap_seconds VALUES (n20+1, n21, NULL, NULL, 21);
        INSERT INTO leap_seconds VALUES (n21+1, n22, NULL, NULL, 22);
```

```
        /* Populating the leap_seconds_tmp table */
         INSERT INTO leap_seconds_tmp VALUES (n0, n1, nominal2true (n0),
                                              nominal2true (n1), 0);
         INSERT INTO leap_seconds_tmp VALUES (n1+1, n2, nominal2true (n1+1),
                                              nominal2true (n2), 1);
         INSERT INTO leap_seconds_tmp VALUES (n2+1, n3, nominal2true (n2+1),
                                              nominal2true (n3), 3);
         INSERT INTO leap_seconds_tmp VALUES (n3+1, n4, nominal2true (n3+1),
                                              nominal2true (n4), 4);
         INSERT INTO leap_seconds_tmp VALUES (n4+1, n5, nominal2true (n4+1),
                                              nominal2true (n5), 5);
         INSERT INTO leap_seconds_tmp VALUES (n5+1, n6, nominal2true (n5+1),
                                              nominal2true (n6), 6);
         INSERT INTO leap_seconds_tmp VALUES (n6+1, n7, nominal2true (n6+1),
                                              nominal2true (n7), 7);
         INSERT INTO leap_seconds_tmp VALUES (n7+1, n8, nominal2true (n7+1),
                                              nominal2true (n8), 8);
         INSERT INTO leap_seconds_tmp VALUES (n8+1, n9, nominal2true (n8+1),
                                              nominal2true (n9), 9);
         INSERT INTO leap_seconds_tmp VALUES (n9+1, n10, nominal2true (n9+1),
                                              nominal2true (n10), 10);
         INSERT INTO leap_seconds_tmp VALUES (n10+1, n11, nominal2true (n10+1),
                                              nominal2true (n11), 11);
         INSERT INTO leap_seconds_tmp VALUES (n11+1, n12, nominal2true (n11+1),
                                              nominal2true (n12), 12);
         INSERT INTO leap_seconds_tmp VALUES (n12+1, n13, nominal2true (n12+1),
                                              nominal2true (n13), 13);
         INSERT INTO leap_seconds_tmp VALUES (n13+1, n14, nominal2true (n13+1),
                                              nominal2true (n14), 14);
         INSERT INTO leap_seconds_tmp VALUES (n14+1, n15, nominal2true (n14+1),
                                              nominal2true (n15), 15);
         INSERT INTO leap_seconds_tmp VALUES (n15+1, n16, nominal2true (n15+1),
                                              nominal2true (n16), 16);
         INSERT INTO leap_seconds_tmp VALUES (n16+1, n17, nominal2true (n16+1),
                                              nominal2true (n17), 17);
         INSERT INTO leap_seconds_tmp VALUES (n17+1, n18, nominal2true (n17+1),
                                              nominal2true (n18), 18);
         INSERT INTO leap_seconds_tmp VALUES (n18+1, n19, nominal2true (n18+1),
                                              nominal2true (n19), 19);
         INSERT INTO leap_seconds_tmp VALUES (n19+1, n20, nominal2true (n19+1),
                                              nominal2true (n20), 20);
         INSERT INTO leap_seconds_tmp VALUES (n20+1, n21, nominal2true (n20+1),
                                              nominal2true (n21), 21);
         INSERT INTO leap_seconds_tmp VALUES (n21+1, n22, nominal2true (n21+1),
                                              nominal2true (n22), 22);
END;
/


 /* Destroying the leap_seconds table */
DROP TABLE leap_seconds;

 /* Creating the leap_seconds table as a copy of the leap_seconds_tmp table */
CREATE TABLE leap_seconds AS SELECT * FROM leap_seconds_tmp;

 /* Destroying the leap_seconds_tmp table */
DROP TABLE leap_seconds_tmp;
```

```
/* Testing string2nominal and string2true functions */

INSERT INTO TEMP VALUES ('1970/01/01 00:00:00', string2nominal ('1970/01/01
00:00:00'), string2true ('1970/01/01 00:00:00'));
INSERT INTO TEMP VALUES ('1972/06/30 23:59:59', string2nominal ('1972/06/30
23:59:59'), string2true ('1972/06/30 23:59:59'));
INSERT INTO TEMP VALUES ('1972/06/30 23:59:60', string2nominal ('1972/06/30
23:59:60'), string2true ('1972/06/30 23:59:60'));
INSERT INTO TEMP VALUES ('1972/07/01 00:00:00', string2nominal ('1972/07/01
00:00:00'), string2true ('1972/07/01 00:00:00'));
INSERT INTO TEMP VALUES ('1972/12/31 23:59:59', string2nominal ('1972/12/31
23:59:59'), string2true ('1972/12/31 23:59:59'));
INSERT INTO TEMP VALUES ('1972/12/31 23:59:60', string2nominal ('1972/12/31
23:59:60'), string2true ('1972/12/31 23:59:60'));
INSERT INTO TEMP VALUES ('1972/12/31 23:59:61', string2nominal ('1972/12/31
23:59:61'), string2true ('1972/12/31 23:59:61'));
INSERT INTO TEMP VALUES ('1973/01/01 00:00:00', string2nominal ('1973/01/01
00:00:00'), string2true ('1973/01/01 00:00:00'));


INSERT INTO TEMP VALUES ('--------------------', NULL, NULL);

 /* Testing nominal2string and nominal2true functions */

INSERT INTO TEMP VALUES (nominal2string (0)       , 0       , nominal2true
(0));
INSERT INTO TEMP VALUES (nominal2string (78796799), 78796799, nominal2true
(78796799));
INSERT INTO TEMP VALUES (nominal2string (78796800), 78796800, nominal2true
(78796800));
INSERT INTO TEMP VALUES (nominal2string (94694399), 94694399, nominal2true
(94694399));
INSERT INTO TEMP VALUES (nominal2string (94694400), 94694400, nominal2true
(94694400));


INSERT INTO TEMP VALUES ('--------------------', NULL, NULL);

 /* Testing true2nominal and true2string functions */

INSERT INTO TEMP VALUES (true2string (0)       , true2nominal (0)       , 0);
INSERT INTO TEMP VALUES (true2string (78796799), true2nominal (78796799),
78796799);
INSERT INTO TEMP VALUES (true2string (78796800), true2nominal (78796800),
78796800);
INSERT INTO TEMP VALUES (true2string (78796801), true2nominal (78796801),
78796801);
INSERT INTO TEMP VALUES (true2string (94694400), true2nominal (94694400),
94694400);
INSERT INTO TEMP VALUES (true2string (94694401), true2nominal (94694401),
94694401);
INSERT INTO TEMP VALUES (true2string (94694402), true2nominal (94694402),
94694402);
INSERT INTO TEMP VALUES (true2string (94694403), true2nominal (94694403),
94694403);
```

```
 /*********************************************************************

SQL> select * from temp;

STRING                   NOMINAL       TRUE
-------------------- ---------- ----------
1970/01/01 00:00:00           0          0
1972/06/30 23:59:59    78796799   78796799
1972/06/30 23:59:60                78796800
1972/07/01 00:00:00    78796800   78796801
1972/12/31 23:59:59    94694399   94694400
1972/12/31 23:59:60                94694401
1972/12/31 23:59:61                94694402
1973/01/01 00:00:00    94694400   94694403
--------------------
1970/01/01 00:00:00           0          0
1972/06/30 23:59:59    78796799   78796799
1972/07/01 00:00:00    78796800   78796801
1972/12/31 23:59:59    94694399   94694400
1973/01/01 00:00:00    94694400   94694403
--------------------
1970/01/01 00:00:00           0          0
1972/06/30 23:59:59    78796799   78796799
1972/06/30 23:59:60                78796800
1972/07/01 00:00:00    78796800   78796801
1972/12/31 23:59:59    94694399   94694400
1972/12/31 23:59:60                94694401
1972/12/31 23:59:61                94694402
1973/01/01 00:00:00    94694400   94694403

23 rows selected.

SQL>

 *********************************************************************/
```

***A***

*afix* ..... 18
*afree* ..... 18
*algorithm* ..... 18
**Amp** ..... 7, 15
*amp1* ..... 20
*amp2* ..... 20
*amp3* ..... 20
*amp4* ..... 20
*amp5* ..... 20
*amp6* ..... 20
*ampid* ..... 19
*amplitude* ..... 19
*ampmeas* ..... 19
*amptype* ..... 20
*arid* ..... 20
**Arrival** ..... 5, 14
**AssocAmM** ..... 8, 16
**AssocAmO** ..... 7, 15
**AssocArO** ..... 6, 15
**AssocCoM** ..... 11, 17
**AssocCoO** ..... 11, 17
*auth* ..... 21
*aziinter* ..... 21
*azilarge* ..... 21
*azimuth* ..... 22
*azismall* ..... 22
*azres* ..... 22

***C***

*channel* ..... 23
*channelsrc* ..... 23
*cmodelid* ..... 23
**Coda** ..... 10, 17
*codatype* ..... 24
*commid* ..... 24

***D***

*datetime* ..... 25
*datumhor* ..... 25
*datumver* ..... 25
*delaz* ..... 26
*delinc* ..... 26
*delslo* ..... 26
*delta* ..... 27
*deltim* ..... 27
*depth* ..... 28
*dip1* ..... 29
*dip2* ..... 29
*dipinter* ..... 28
*diplarge* ..... 28
*dipsmall* ..... 29
*distance* ..... 30
*durtype* ..... 30

***E***

*eigenn* ..... 30
*eigenp* ..... 31
*eigent* ..... 31
*ema* ..... 31
*emares* ..... 32
*eramp* ..... 32
*erhor* ..... 32
*erlat* ..... 33
*erlon* ..... 33
*erscalar* ..... 33
*etype* ..... 34
**Event** ..... 3, 13
*evid* ..... 34
*evname* ..... 35
*exp* ..... 35

***F***

*fdepth* ..... 35
*fepi* ..... 36
*flagamp* ..... 36
*fm.* ..... 37
*ftime* ..... 37

***G***

*gap* ..... 38

***I***

*importance* ..... 38
*iphase* ..... 38

***L***

*lat.* ..... 39
*lddate* ..... 39
*lineno* ..... 39
*locevid* ..... 40
*lon* ..... 40

***M***

*mag* ..... 40
mag*algo* ..... 41
*magcorr* ..... 41
*magid* ..... 41
*maginter* ..... 42
*maglarge* ..... 42
*magnitude* ..... 42
*magres* ..... 43
*magsmall* ..... 43
*magtype* ..... 44
**Mec** ..... 8, 16
*mecalgo* ..... 45
*mechtype* ..... 45

*mecid* ..... 24, 44
*mmi* ..... 45
*mxx* ..... 46
*mxy* ..... 46
*mxz* ..... 46
*myy* ..... 46
*myz* ..... 46
*mzz* ..... 46

***N***

*nbp* ..... 46
*nbs* ..... 46
*ndef* ..... 47
*net* ..... 47
**Netmag** ..... 5, 14
*nfelt* ..... 47
*nsample* ..... 48
*nsta* ..... 48

***O***

*orid* ..... 48
*oridin* ..... 49
*oridout* ..... 49
**Origin** ..... 3, 14

***P***

*pclvd* ..... 52
*pdc* ..... 52
*per* ..... 49
*pga* ..... 50
*piso* ..... 53
*plungen* ..... 50
*plungep* ..... 50
*plunget* ..... 51
*prefmag* ..... 51
*prefmec* ..... 51
*prefor* ..... 52
*pvr* ..... 53

***Q***

*qfix* ..... 53
*qfree* ..... 54
*qual* ..... 54
*quality* ..... 54

***R***

*rake1* ..... 55
*rake2* ..... 55
*remark* ..... 55
**Remark** ..... 8, 16
*remarks* ..... 56
*rms* ..... 56

***S***

*scalar* ..... 56
*scorr* ..... 57
*sdelay* ..... 57
*sdepth* ..... 57
*sdobs* ..... 58
*seaz* ..... 58
*seedchan* ..... 59
**Significant_Event** ..... 3, 13
*slores* ..... 60
*slow* ..... 60
*snr* ..... 60
*srcduration* ..... 61
*sta* ..... 61
*stime* ..... 61
*strike1* ..... 63
*strike2* ..... 63
*striken* ..... 62
*strikep* ..... 62
*striket* ..... 62
*stt.* ..... 64
*stx.* ..... 64
*sty.* ..... 64
*stz.* ..... 64
*subsource* ..... 64
*sxx* ..... 64
*sxy* ..... 64
*sxz* ..... 64
*syy* ..... 64
*syz* ..... 64
*szz.* ..... 64

***T***

*tau* ..... 65
*tfd.* ..... 65
*tft* ..... 65
*time1* ..... 66
*time2* ..... 66
*time3* ..... 66
*time4* ..... 66
*time5* ..... 66
*time6* ..... 66
*timeres* ..... 66
*totalamp* ..... 66
*totalarr* ..... 67
*type* ..... 67

***U***

*uncertainty* ..... 67
*undip1* ..... 68
*undip2* ..... 68
*units* ..... 69
*unmxx* ..... 69
*unmxy* ..... 69
*unmxz* ..... 69
*unmyy* ..... 69
*unmyz* ..... 69
*unmzz* ..... 69
*unrake1* ..... 70

*unrake2* ..... 70
*unstrike1* ..... 70
*unstrike2* ..... 71

***V***

*vmodelid* ..... 71

***W***

*weight* ..... 71
*wgt* ..... 72