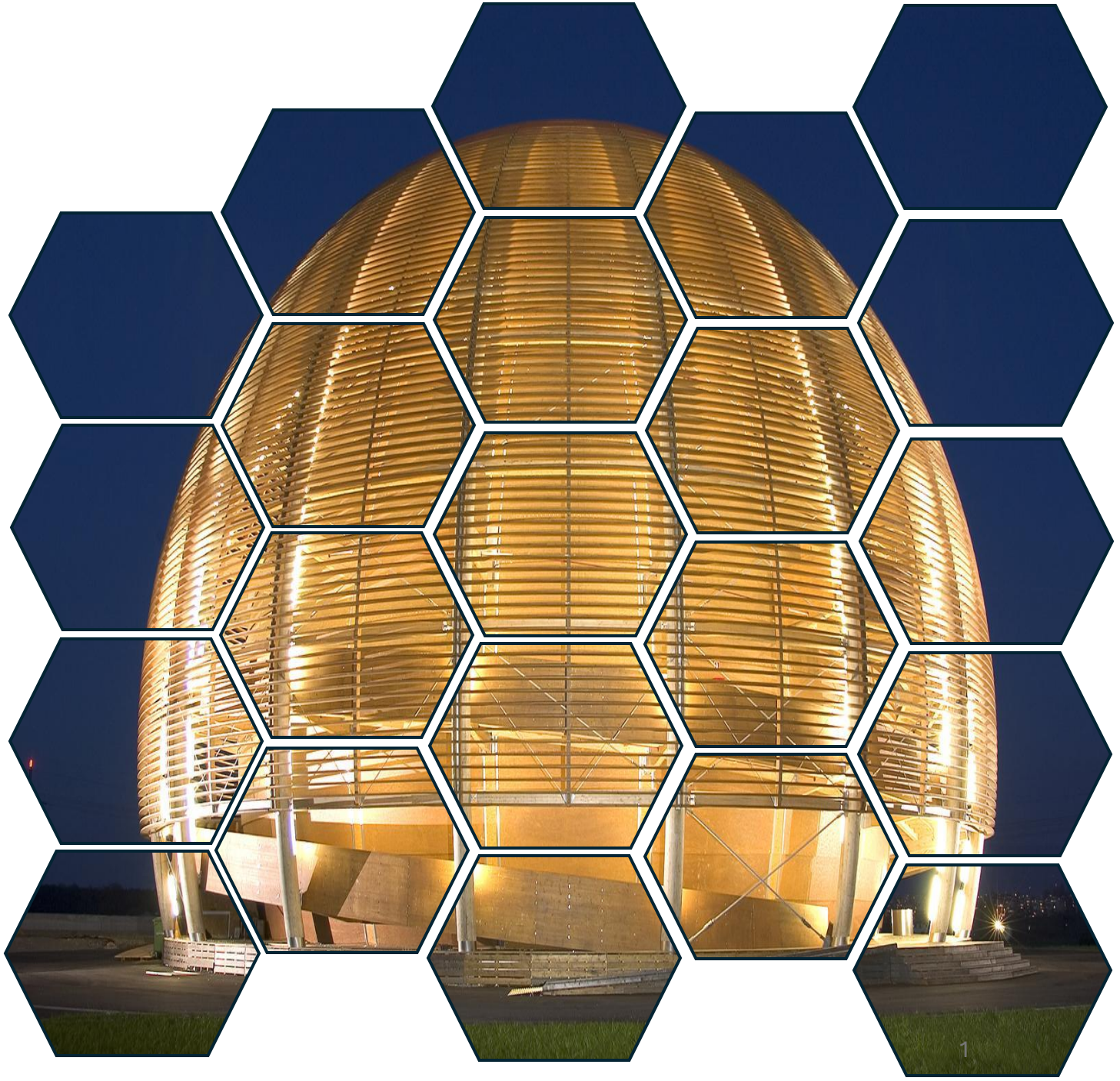
O2JE tutorial

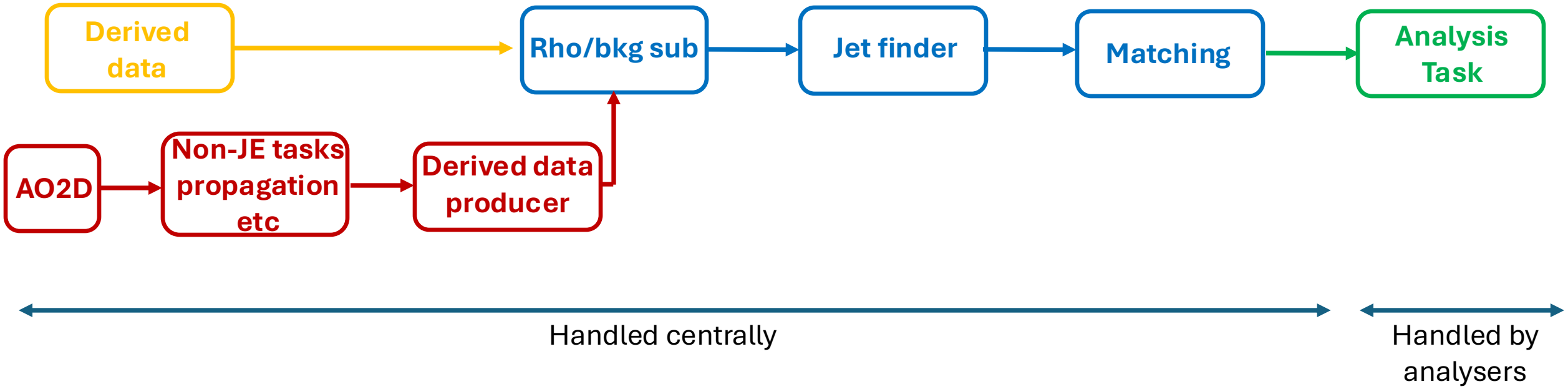
Nima Zardoshti (CERN)

O2 analysis tutorial 5.0
13/11/2025



The jet derived-data model

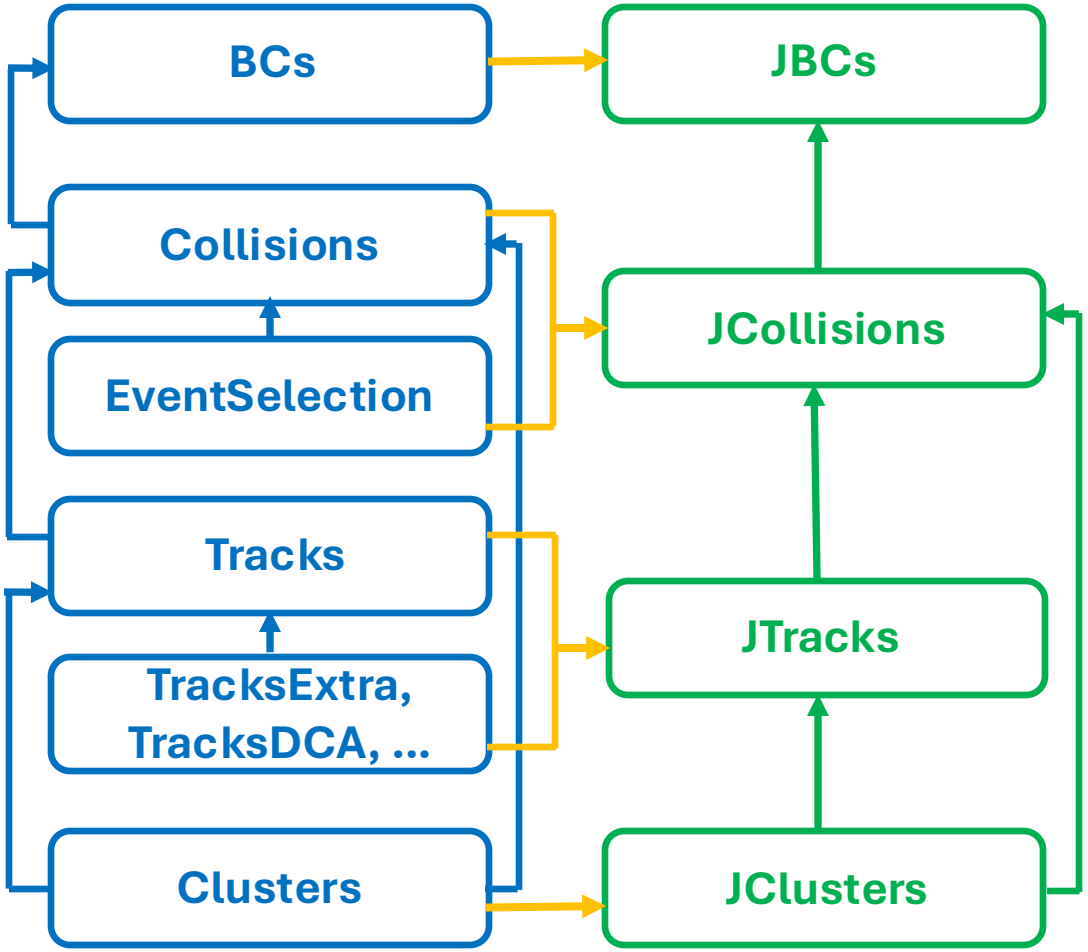
- The jet framework works exclusively on the JE derived format – this means that for most analyses the original AO2D tables and setup tasks are not required



- For analyses that require it, the use of non JE tables is possible. However, this is only allowed if a given table cannot be added to the JE derived data format (such as PID). These analyses should use JE tables where possible and non-JE tables only where required

The jet derived-data model

- The JE derived data format is composed of reduced versions of tables commonly needed for jet finding – lives in the PWGJE folder of O2Physics



Not all tables are shown but this is the basic concept

non-derived tables

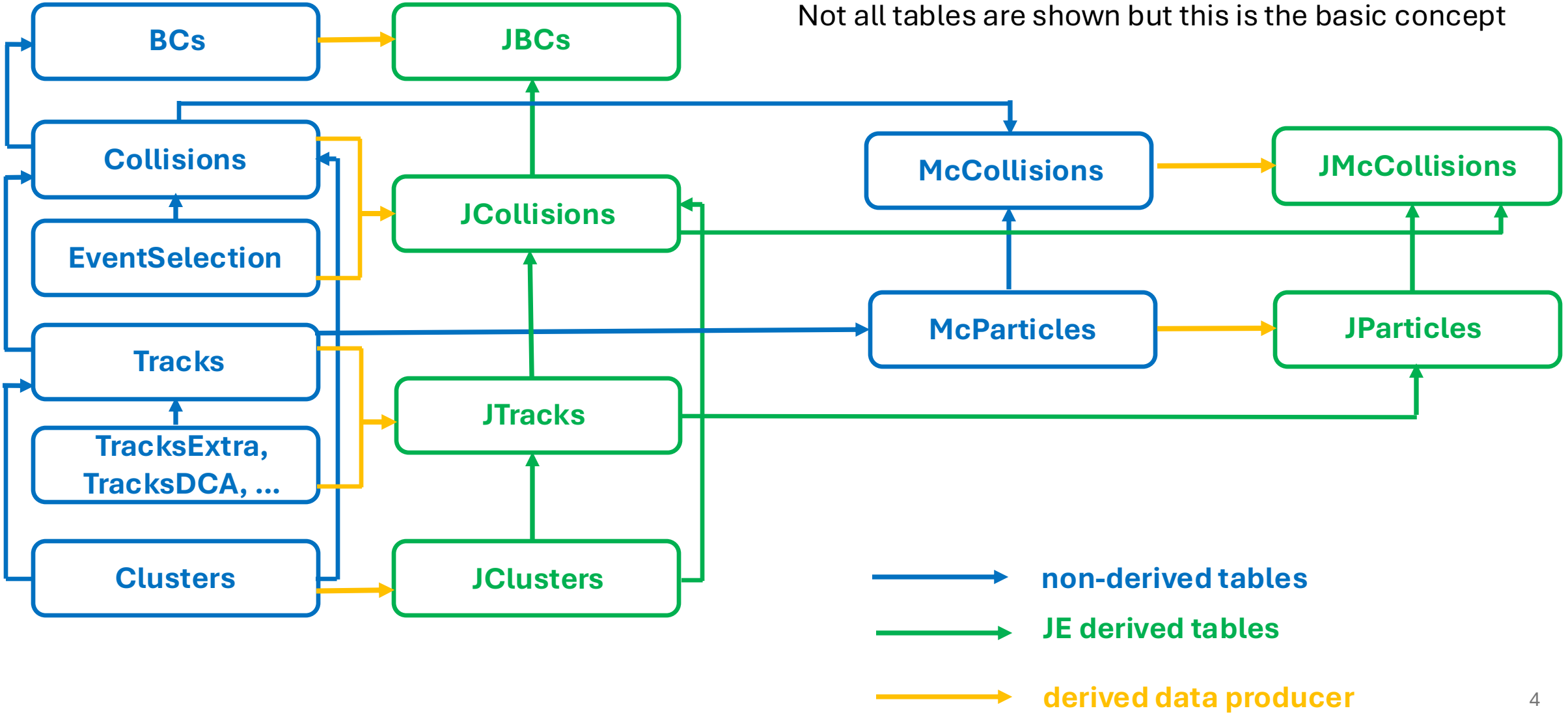
JE derived tables

derived data producer

The jet derived-data model

- The JE derived data format is composed of reduced versions of tables commonly needed for jet finding – lives in the PWGJE folder of O2Physics

Not all tables are shown but this is the basic concept



Example of JTables

JCollisions table

```
DECLARE_SOA_TABLE_STAGED(JCollisions, "JCOLLISION",
                         o2::soa::Index<>,
                         jcollision::PosX,
                         jcollision::PosY,
                         jcollision::PosZ,
                         jcollision::MultFV0A,
                         jcollision::MultFV0C,
                         jcollision::MultFV0M<jcollision::MultFV0A, jcollision::MultFV0C>,
                         jcollision::MultFT0A,
                         jcollision::MultFT0C,
                         jcollision::MultFT0M<jcollision::MultFT0A, jcollision::MultFT0C>,
                         jcollision::CentFV0A,
                         jcollision::CentFV0M,
                         jcollision::CentFT0A,
                         jcollision::CentFT0C,
                         jcollision::CentFT0M,
                         jcollision::CentFT0CVariant1,
                         jcollision::HadronicRate,
                         jcollision::TrackOccupancyInTimeRange,
                         jcollision::Alias,
                         jcollision::EventSel,
                         jcollision::Rct,
                         jcollision::TriggerSel);
```

Base event table

Quantities from intermediate tasks

Flags from selection task

JTracks table

```
DECLARE_SOA_TABLE_STAGED(JTracks, "JTRACK",
                         o2::soa::Index<>,
                         jtrack::JCollisionId,
                         jtrack::Pt,
                         jtrack::Eta,
                         jtrack::Phi,
                         jtrack::TrackSel,
                         jtrack::Px<jtrack::Pt, jtrack::Phi>,
                         jtrack::Py<jtrack::Pt, jtrack::Phi>,
                         jtrack::Pz<jtrack::Pt, jtrack::Eta>,
                         jtrack::P<jtrack::Pt, jtrack::Eta>,
                         jtrack::Energy<jtrack::Pt, jtrack::Eta>,
                         jtrack::Sign<jtrack::TrackSel>);
```

Link to JCollisions

Base track table

Flags from selection task

Example of JTables

JCollisions table

```
DECLARE_SOA_TABLE_STAGED(JCollisions, "JCOLLISION",
                         o2::soa::Index<>,
                         jcollision::PosX,
                         jcollision::PosY,
                         jcollision::PosZ,
                         jcollision::MultFV0A,
                         jcollision::MultFV0C,
                         jcollision::MultFV0M<jcollision::MultFV0A, jcollision::MultFV0C>,
                         jcollision::MultFT0A,
                         jcollision::MultFT0C,
                         jcollision::MultFT0M<jcollision::MultFT0A, jcollision::MultFT0C>,
                         jcollision::CentFV0A,
                         jcollision::CentFV0M,
                         jcollision::CentFT0A,
                         jcollision::CentFT0C,
                         jcollision::CentFT0M,
                         jcollision::CentFT0CVariant1,
                         jcollision::HadronicRate,
                         jcollision::TrackOccupancyInTimeRange,
                         jcollision::Alias,
                         jcollision::EventSel,
                         jcollision::Rct,
                         jcollision::TriggerSel);
```

Base event table

Quantities from intermediate tasks

Flags from selection task

JTracks table

```
DECLARE_SOA_TABLE_STAGED(JTracks, "JTRACK",
                         o2::soa::Index<>,
                         jtrack::JCollisionId,
                         jtrack::Pt,
                         jtrack::Eta,
                         jtrack::Phi,
                         jtrack::TrackSel,
                         jtrack::Px<jtrack::Pt, jtrack::Phi>,
                         jtrack::Py<jtrack::Pt, jtrack::Phi>,
                         jtrack::Pz<jtrack::Pt, jtrack::Eta>,
                         jtrack::P<jtrack::Pt, jtrack::Eta>,
                         jtrack::Energy<jtrack::Pt, jtrack::Eta>,
                         jtrack::Sign<jtrack::TrackSel>);
```

Link to JCollisions

Base track table

Flags from selection task

```
DECLARE_SOA_TABLE_STAGED(JCollisionPIs, "JCOLLISIONPI",
                         jcollision::CollisionId);
```

Tables with the PI suffix link back to non-derived tables

Example of JTables

JCollisions table

```
DECLARE_SOA_TABLE_STAGED(JCollisions, "JCOLLISION",
                         o2::soa::Index<>,
                         jcollision::PosX,
                         jcollision::PosY,
                         jcollision::PosZ,
                         jcollision::MultFV0A,
                         jcollision::MultFV0C,
                         jcollision::MultFV0M<jcollision::MultFV0A, jcollision::MultFV0C>,
                         jcollision::MultFT0A,
                         jcollision::MultFT0C,
                         jcollision::MultFT0M<jcollision::MultFT0A, jcollision::MultFT0C>,
                         jcollision::CentFV0A,
                         jcollision::CentFV0M,
                         jcollision::CentFT0A,
                         jcollision::CentFT0C,
                         jcollision::CentFT0M,
                         jcollision::CentFT0CVariant1,
                         jcollision::HadronicRate,
                         jcollision::TrackOccupancyInTimeRange,
                         jcollision::Alias,
                         jcollision::EventSel,
                         jcollision::Rct,
                         jcollision::TriggerSel);
```

Base event table

Quantities from intermediate tasks

Flags from selection task

JTracks table

```
DECLARE_SOA_TABLE_STAGED(JTracks, "JTRACK",
                         o2::soa::Index<>,
                         jtrack::JCollisionId,
                         jtrack::Pt,
                         jtrack::Eta,
                         jtrack::Phi,
                         jtrack::TrackSel,
                         jtrack::Px<jtrack::Pt, jtrack::Phi>,
                         jtrack::Py<jtrack::Pt, jtrack::Phi>,
                         jtrack::Pz<jtrack::Pt, jtrack::Eta>,
                         jtrack::P<jtrack::Pt, jtrack::Eta>,
                         jtrack::Energy<jtrack::Pt, jtrack::Eta>,
                         jtrack::Sign<jtrack::TrackSel>);
```

Link to JCollisions

Base track table

Flags from selection task

Derived tables are stored in DataModel/JetReducedData.h

Flags such as selections in the derived format are defined in Core/JetDerivedDataUtilities.h

```
DECLARE_SOA_TABLE_STAGED(JCollisionPIs, "JCOLLISIONPI",
                         jcollision::CollisionId);
```

Tables with the PI suffix link back to non-derived tables

Derived tables are grouped by keywords in DataModel/Jet.h these keywords should always be used

```
using JetTracks = JTracks;
using JetTracksMCD = o2::soa::Join<JetTracks, JMcTrackLbs>;
```

Additional tables available commonly in the framework

Filled by jetFinder tasks

```
DECLARE_SOA_TABLE(_jet_type_##s, "AOD", _description_,
                  o2::soa::Index<>,
                  jet::_collision_name_##Id,
                  jet::Pt,
                  jet::Eta,
                  jet::Phi,
                  jet::Energy,
                  jet::Y,
                  jet::Mass,
                  jet::Area,
                  jet::R,
                  jet::Px<jet::Pt, jet::Phi>,
                  jet::Py<jet::Pt, jet::Phi>,
                  jet::Pz<jet::Pt, jet::Eta>,
                  jet::P<jet::Pt, jet::Eta>,
                  _name_##util::Dummy##_jet_type_##s<>);
```

Jet Table

Jet tables have a row filled for each jet
Different tables for Data, MCD (detector-level MC) and MCP (particle-level MC) jets
Different jet radii go into the same table

```
DECLARE_JET_TABLES_LEVELS(Charged, JTrackSub, HfD0Bases, HfD0PBases, "C");
DECLARE_JET_TABLES_LEVELS(Full, JTrackSub, HfD0Bases, HfD0PBases, "F");
DECLARE_JET_TABLES_LEVELS(Neutral, JTrackSub, HfD0Bases, HfD0PBases, "N");
DECLARE_JET_TABLES_LEVELS(D0Charged, JTrackD0Sub, HfD0Bases, HfD0PBases, "D0");
DECLARE_JET_TABLES_LEVELS(DplusCharged, JTrackDplusSub, HfDplusBases, HfDplusPBases, "DP");
DECLARE_JET_TABLES_LEVELS(LcCharged, JTrackLcSub, HfLcBases, HfLcPBases, "Lc");
DECLARE_JET_TABLES_LEVELS(BplusCharged, JTrackBplusSub, HfBplusBases, HfBplusPBases, "BP");
DECLARE_JET_TABLES_LEVELS(V0Charged, JTrackSub, V0Cores, JV0Mcs, "V0");
DECLARE_JET_TABLES_LEVELS(DielectronCharged, JTrackSub, Dielectrons, JDielectronMcs, "DIEL");
```

Filled by jetFinder tasks

```
DECLARE_SOA_TABLE(_jet_type_##Constituents, "AOD", _Description_ "C",
                  _name_##constituents::_jet_type_##Id,
                  _name_##constituents::_track_type_##Ids,
                  _name_##constituents::JClusterIds,
                  _name_##constituents::CandidatesIds);
```

Jet Constituents Table

Jet constituents table points to particular rows in the relevant tables.
JTracks, JClusters, etc

Filled by jetMatching tasks

```
DECLARE_SOA_TABLE(_jet_type_base_##JetsMatchedTo##_jet_type_tag_##Jets, "AOD", _description_,
                  _jet_type_tag_##jetmatchingGeo::_jet_type_tag_##JetIds,
                  _jet_type_tag_##jetmatchingPt::_jet_type_tag_##JetIds,
                  _jet_type_tag_##jetmatchingCand::_jet_type_tag_##JetIds,
                  _jet_type_base_##jetsmatchedto##_jet_type_tag_::Dummy##_jet_type_base_##s<>);
```

MCD and MCP jets and Subtracted and Unsubtracted jets are matched

Jet Matching Table

Additional tables available commonly in the framework

Filled by rhoEstimator tasks

```
DECLARE_SOA_TABLE(BkgChargedRhos, "AOD", "BkgCRho",
                  o2::soa::Index<>,
                  bkgrho::Rho,
                  bkgrho::RhoM);

DECLARE_SOA_TABLE(BkgChargedMcRhos, "AOD", "BkgCMcRho",
                  o2::soa::Index<>,
                  bkgrho::Rho,
                  bkgrho::RhoM,
                  o2::soa::Marker<1>);
```

Provides multiple ways of estimating the median background density per event

Filled by EventWiseConstituentSubtractor task

```
DECLARE_SOA_TABLE(JTrackSubs, "AOD", "JTrackSubs",
                  o2::soa::Index<>,
                  bkgcharged::JCollisionId,
                  jtrack::Pt,
                  jtrack::Eta,
                  jtrack::Phi,
                  jtrack::TrackSel,
                  jtrack::Px<jtrack::Pt, jtrack::Phi>,
                  jtrack::Py<jtrack::Pt, jtrack::Phi>,
                  jtrack::Pz<jtrack::Pt, jtrack::Eta>,
                  jtrack::P<jtrack::Pt, jtrack::Eta>,
                  jtrack::Energy<jtrack::Pt, jtrack::Eta>);

ıg JTrackSub = JTrackSubs::iterator;

_ARE_SOA_TABLE(JMcParticleSubs, "AOD", "JMcPartSubs",
                  o2::soa::Index<>,
                  bkgcharged::JMcCollisionId,
                  jmcparticle::Pt,
                  jmcparticle::Eta,
                  jmcparticle::Phi,
                  jmcparticle::Y,
                  jmcparticle::E,
                  jmcparticle::PdgCode,
                  jmcparticle::StatusCode,
                  jmcparticle::Flags,
                  jmcparticle::Px<jmcparticle::Pt, jmcparticle::Phi>,
                  jmcparticle::Py<jmcparticle::Pt, jmcparticle::Phi>,
                  jmcparticle::Pz<jmcparticle::Pt, jmcparticle::Eta>,
                  jmcparticle::P<jmcparticle::Pt, jmcparticle::Eta>,
                  jmcparticle::Energy<jmcparticle::E>,
                  jmcparticle::ProducedByGenerator<jmcparticle::Flags>,
                  jmcparticle::FromBackgroundEvent<jmcparticle::Flags>,
                  jmcparticle::GetProcess<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::GetGenStatusCode<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::GetHepMCStatusCode<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::IsPhysicalPrimary<jmcparticle::Flags>);

using JMcParticleSub = JMcParticleSubs::iterator;
```

Provides separate subtracted track and particle tables that can be used in jet finding and analyses

Additional tables available commonly in the framework

Filled by rhoEstimator tasks

```
DECLARE_SOA_TABLE(BkgChargedRhos, "AOD", "BkgCRho",
                  o2::soa::Index<>,
                  bkgrho::Rho,
                  bkgrho::RhoM);

DECLARE_SOA_TABLE(BkgChargedMcRhos, "AOD", "BkgCMcRho",
                  o2::soa::Index<>,
                  bkgrho::Rho,
                  bkgrho::RhoM,
                  o2::soa::Marker<1>);
```

Provides multiple ways of estimating the median background density per event

More useful tables exist, please take a look

Filled by EventWiseConstituentSubtractor task

```
DECLARE_SOA_TABLE(JTrackSubs, "AOD", "JTrackSubs",
                  o2::soa::Index<>,
                  bkgcharged::JCollisionId,
                  jtrack::Pt,
                  jtrack::Eta,
                  jtrack::Phi,
                  jtrack::TrackSel,
                  jtrack::Px<jtrack::Pt, jtrack::Phi>,
                  jtrack::Py<jtrack::Pt, jtrack::Phi>,
                  jtrack::Pz<jtrack::Pt, jtrack::Eta>,
                  jtrack::P<jtrack::Pt, jtrack::Eta>,
                  jtrack::Energy<jtrack::Pt, jtrack::Eta>);

ıg JTrackSub = JTrackSubs::iterator;

_ARE_SOA_TABLE(JMcParticleSubs, "AOD", "JMcPartSubs",
                  o2::soa::Index<>,
                  bkgcharged::JMcCollisionId,
                  jmcparticle::Pt,
                  jmcparticle::Eta,
                  jmcparticle::Phi,
                  jmcparticle::Y,
                  jmcparticle::E,
                  jmcparticle::PdgCode,
                  jmcparticle::StatusCode,
                  jmcparticle::Flags,
                  jmcparticle::Px<jmcparticle::Pt, jmcparticle::Phi>,
                  jmcparticle::Py<jmcparticle::Pt, jmcparticle::Phi>,
                  jmcparticle::Pz<jmcparticle::Pt, jmcparticle::Eta>,
                  jmcparticle::P<jmcparticle::Pt, jmcparticle::Eta>,
                  jmcparticle::Energy<jmcparticle::E>,
                  jmcparticle::ProducedByGenerator<jmcparticle::Flags>,
                  jmcparticle::FromBackgroundEvent<jmcparticle::Flags>,
                  jmcparticle::GetProcess<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::GetGenStatusCode<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::GetHepMCStatusCode<jmcparticle::Flags, jmcparticle::StatusCode>,
                  jmcparticle::IsPhysicalPrimary<jmcparticle::Flags>);

using JMcParticleSub = JMcParticleSubs::iterator;
```

Provides separate subtracted track and particle tables that can be used in jet finding and analyses

Examples of how to use the framework

Now we can hop onto the hands on session and look at how to build up an analysis using the jetTutorial task

A few notes:

Analysis tasks should be committed to the Tasks folder. Any tasks in this folder are individual tasks and are not meant for common use. Tasks in the JetFinders, JetMatching and TableProducer folders are meant for common use

Unless not possible your tasks should only subscribe to Jet derived tables. This will allow you to run over the derived data formats. If this is not possible please get in touch before starting work on your task to discuss options

For analysers interested in working on O2 please join the following mattermost channels :
PWGJE O2 (for all JE O2 related queries)
JE O2 MC (for testing and updates on JE MC)
O2 Analysis (for general queries not related to the jet framework)
O2 Analysis Announcements (for new non-JE feature announcements in O2)
O2 Hyperloop Operations (for requesting the submission of trains)

Once you have your first task committed you can create your analysis page on hyperloop. You will find many common wagons in Service Wagons JE which you should use where possible