

[Introduction](#)[Welcome](#)[Access](#)[API Keys](#)[Tutorials](#)[About](#)[Operations](#)[Queries](#)[Subscriptions](#)[Types](#)[AdjustedPlaye...](#)[AdjustedPlaye...](#)[AdjustedPlaye...](#)[AdjustedPlaye...](#)[AdjustedPlaye...](#)[AdjustedPlaye...](#)

CFBD GraphQL API

Welcome to CFBD GraphQL API! This API provides mechanisms for querying and subscribing to the CFBD dataset using GraphQL. Read below for more information.

Contact

API Support

admin@collegefootballdata.com

<https://collegefootballdata.com/about>

Terms of Service

<https://collegefootballdata.com/about>

API ENDPOINTS

Production:

<https://graphql.collegefootballdata.com>

HEADERS

Authorization: Bearer <YOUR_TOKEN_H

Access

You must be subscribed to Patreon Tier 3 or higher to access the GraphQL API. Visit the [CFBD Patreon page](#) to subscribe.

AdjustedPlaye...
AdjustedPlaye...
AdjustedPlaye...
AdjustedPlaye...
AdjustedPlaye...
AdjustedPlaye...

API Keys

You can use the same API key you use for the REST API to access the GraphQL API. The key should be passed in the Authorization header as a Bearer token. Visit <https://collegefootballdata.com/key> to register a new key. A key should be autogenerated and emailed to you if no key is associated with your email when you subscribe to the Patreon. It can take up to 15 minutes for Patreon benefits and access to sync up.

Tutorials

A [general tutorial](#) can be found on the CFBD Blog. A more in-depth [tutorial on subscriptions](#) can also be found on the Blog.

About

This API offers a more dynamic way to query the CFBD dataset using GraphQL queries. It also provides a mechanism for realtime data updates using GraphQL subscriptions. Note that not all data is

currently supported in this API. View these docs for a list of available queries and subscriptions.

Queries

adjustedPlayerMetrics

An array relationship

Response

Returns `[AdjustedPlayerMetrics!]!`

Arguments

Name	Description
<code>distinctOn - [AdjustedPlayerMetrics!]</code>	distinct select on <code>columns[selectColumn!]</code>
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>

QUERY

```
query AdjustedPlayerMetrics(  
  $distinctOn: [AdjustedPlayerMetri  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [AdjustedPlayerMetricsOrder  
  $where: AdjustedPlayerMetricsBool  
) {  
  adjustedPlayerMetrics(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    athlete {  
      adjustedPlayerMetrics {  
        ...AdjustedPlayerMetricsFra  
      }  
      adjustedPlayerMetricsAggregat
```

Name	Description
orderBy - [AdjustedPlayerMetricsOrderBy!]	sort the rows by one or more columns
where - AdjustedPlayerMetricsWhereExp	filter the rows

```

    ...AdjustedPlayerMetricsAgg
}
athleteTeams {
    ...AthleteTeamFragment
}
athleteTeamsAggregate {
    ...AthleteTeamAggregateFrag
}
firstName
height
hometown {
    ...HometownFragment
}
hometownId
id
jersey
lastName
name
position {
    ...PositionFragment
}
positionId
recruits {
    ...RecruitFragment
}
recruitsAggregate {
    ...RecruitAggregateFragment
}
teamId
weight
}
athleteId
metricType
metricValue
plays
year

```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["athleteId"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [AdjustedPlayerMetrics  
  "where": AdjustedPlayerMetricsBoo  
}  
[REDACTED]
```

RESPONSE

```
{  
  "data": {  
    "adjustedPlayerMetrics": [  
      {  
        "athlete": Athlete,  
        "athleteId": bigint,  
        "metricType": player_adjust,  
        "metricValue": numeric,  
        "plays": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}[REDACTED]
```

Queries

adjustedPlayerMetricsAggregate

An aggregate relationship

Response

Returns an
[AdjustedPlayerMetricsAggregate!](#)

Arguments

Name	Description
distinctOn - [AdjustedPlayerMetrics]!<sub>columns</sub>	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AdjustedPlayerMetricsOrderBy!]!	sort the rows by one or more columns

QUERY

```
query AdjustedPlayerMetricsAggregate {
  distinctOn: [AdjustedPlayerMetrics]!
  limit: Int,
  offset: Int,
  orderBy: [AdjustedPlayerMetricsOrder]
  where: AdjustedPlayerMetricsBool
} {
  adjustedPlayerMetricsAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...AdjustedPlayerMetricsAvg
      }
      count
      max {
        ...AdjustedPlayerMetricsMax
      }
      min {
        ...AdjustedPlayerMetricsMin
      }
      stdDev {
        ...AdjustedPlayerMetricsStdDev
      }
    }
  }
}
```

Name	Description
where -	filter the rows
AdjustedPlayerMetrics	return <code>None</code>

```
    ...AdjustedPlayerMetricsStd
}
stddevPop {
    ...AdjustedPlayerMetricsStd
}
stddevSamp {
    ...AdjustedPlayerMetricsStd
}
sum {
    ...AdjustedPlayerMetricsSum
}
varPop {
    ...AdjustedPlayerMetricsVar
}
varSamp {
    ...AdjustedPlayerMetricsVar
}
variance {
    ...AdjustedPlayerMetricsVar
}
}
nodes {
    athlete {
        ...AthleteFragment
    }
    athleteId
    metricType
    metricValue
    plays
    year
}
```

VARIABLES

```
{  
  "distinctOn": ["athleteId"],
```

```
        "limit": 123,  
        "offset": 987,  
        "orderBy": [AdjustedPlayerMetrics  
      "where": AdjustedPlayerMetricsBoo  
    }]
```

RESPONSE

```
{  
  "data": {  
    "adjustedPlayerMetricsAggregate":  
      "aggregate": AdjustedPlayerMe  
      "nodes": [AdjustedPlayerMetri  
    }  
  }  
}
```



Queries

adjustedTeamMetrics

fetch data from the table:
"adjusted_team_metrics"

Response

Returns [AdjustedTeamMetrics!]!

Arguments

Name	Description
distinctOn - [AdjustedTeamMetricsColumn!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AdjustedTeamMetricsOrderBy!]	sort the rows by one or more columns
where - AdjustedTeamMetricsFilter!	filter the rows

QUERY

```
query AdjustedTeamMetrics(  
    $distinctOn: [AdjustedTeamMetrics!]!,  
    $limit: Int!,  
    $offset: Int!,  
    $orderBy: [AdjustedTeamMetricsOrdering!]!,  
    $where: AdjustedTeamMetricsBooleanExpres-  
) {  
  adjustedTeamMetrics(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
  epa  
  epaAllowed  
  explosiveness  
  explosivenessAllowed  
  highlightYards  
  highlightYardsAllowed  
  lineYards  
  lineYardsAllowed  
  openFieldYards  
  openFieldYardsAllowed  
  passingDownsSuccess  
  passingDownsSuccessAllowed  
  passingEpa  
  passingEpaAllowed  
  rushingEpa  
  rushingEpaAllowed  
  secondLevelYards  
  secondLevelYardsAllowed  
  standardDownsSuccess  
  standardDownsSuccessAllowed  
  success  
  successAllowed  
  team {
```

```
        abbreviation
        classification
        conference
        conferenceId
        division
        school
        teamId
    }
    teamId
    year
}
}
```

VARIABLES

```
{
  "distinctOn": ["epa"],
  "limit": 123,
  "offset": 123,
  "orderBy": [AdjustedTeamMetricsOr
  "where": AdjustedTeamMetricsBoole
}
```

RESPONSE

```
{
  "data": {
    "adjustedTeamMetrics": [
      {
        "epa": numeric,
        "epaAllowed": numeric,
        "explosiveness": numeric,
        "explosivenessAllowed": num
        "highlightYards": numeric,
        "highlightYardsAllowed": nu
```

```
        "lineYards": numeric,
        "lineYardsAllowed": numeric
      "openFieldYards": numeric,
      "openFieldYardsAllowed": nu
    "passingDownsSuccess": num
    "passingDownsSuccessAllowed"
    "passingEpa": numeric,
    "passingEpaAllowed": numeri
    "rushingEpa": numeric,
    "rushingEpaAllowed": numeri
    "secondLevelYards": numeric
    "secondLevelYardsAllowed": 
  "standardDownsSuccess": num
  "standardDownsSuccessAllowe
  "success": numeric,
  "successAllowed": numeric,
  "team": currentTeams,
  "teamId": 987,
  "year": smallint
}
]
}
}
```

Queries

adjustedTeamMetricsAggregate

fetch aggregated fields from the table:
 "adjusted_team_metrics"

Response

Returns an
[AdjustedTeamMetricsAggregate!](#)

Arguments

Name	Description
distinctOn - [AdjustedTeamMetricsColumns!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AdjustedTeamMetricsOrderBy!]	sort the rows by one or more columns
where - AdjustedTeamMetricsWhereInput	filter the rows

QUERY

```
query AdjustedTeamMetricsAggregate(
  $distinctOn: [AdjustedTeamMetrics
    $limit: Int,
    $offset: Int,
    $orderBy: [AdjustedTeamMetricsOrd
    $where: AdjustedTeamMetricsBoolEx
  ) {
    adjustedTeamMetricsAggregate(
      distinctOn: $distinctOn,
      limit: $limit,
      offset: $offset,
      orderBy: $orderBy,
      where: $where
    ) {
      aggregate {
        avg {
          ...AdjustedTeamMetricsAvgFi
        }
        count
        max {
          ...AdjustedTeamMetricsMaxFi
        }
        min {
          ...AdjustedTeamMetricsMinFi
        }
        stdDev {
          ...AdjustedTeamMetricsStdde
        }
        stdDevPop {
          ...AdjustedTeamMetricsStdde
        }
        stdDevSamp {
          ...AdjustedTeamMetricsStdde
        }
      }
    }
  }
)
```

```
        }
      sum {
        ...AdjustedTeamMetricsSumFragment
      }
      varPop {
        ...AdjustedTeamMetricsVarPopFragment
      }
      varSamp {
        ...AdjustedTeamMetricsVarSampFragment
      }
      variance {
        ...AdjustedTeamMetricsVarianceFragment
      }
    }
  nodes {
    epa
    epaAllowed
    explosiveness
    explosivenessAllowed
    highlightYards
    highlightYardsAllowed
    lineYards
    lineYardsAllowed
    openFieldYards
    openFieldYardsAllowed
    passingDownsSuccess
    passingDownsSuccessAllowed
    passingEpa
    passingEpaAllowed
    rushingEpa
    rushingEpaAllowed
    secondLevelYards
    secondLevelYardsAllowed
    standardDownsSuccess
    standardDownsSuccessAllowed
    success
    successAllowed
    team {
      ...currentTeamsFragment
    }
  }
}
```

```
        teamId  
        year  
    }  
}  
}
```

VARIABLES

```
{  
  "distinctOn": ["epa"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [AdjustedTeamMetricsOr  
  "where": AdjustedTeamMetricsBoole  
}
```

RESPONSE

```
{  
  "data": {  
    "adjustedTeamMetricsAggregate":  
      "aggregate": AdjustedTeamMet  
      "nodes": [AdjustedTeamMetrics  
    }  
  }  
}
```

athlete

fetch data from the table: "athlete"

Response

Returns `[Athlete!]!`

Arguments

Name	Description
<code>distinctOn - [AthleteSelectColumn]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [AthleteOrderBy!]</code>	sort the rows by one or more columns
<code>where - AthleteBoolExp</code>	filter the rows returned

QUERY

```
query Athlete(
  $distinctOn: [AthleteSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteOrderBy!],
  $where: AthleteBoolExp
) {
  athlete(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    adjustedPlayerMetrics {
      athlete {
        ...AthleteFragment
      }
      athleteId
      metricType
      metricValue
      plays
      year
    }
    adjustedPlayerMetricsAggregate {
      aggregate {
        ...AdjustedPlayerMetricsAgg
      }
    }
    nodes {
      ...
    }
  }
}
```

```
    ...AdjustedPlayerMetricsFra
  }
}
athleteTeams {
  athlete {
    ...AthleteFragment
  }
  athleteId
  endYear
  startYear
  team {
    ...historicalTeamFragment
  }
  teamId
}
athleteTeamsAggregate {
  aggregate {
    ...AthleteTeamAggregateField
  }
  nodes {
    ...AthleteTeamFragment
  }
}
firstName
height
hometown {
  athletes {
    ...AthleteFragment
  }
  athletesAggregate {
    ...AthleteAggregateFragment
  }
  city
  country
  countyFips
  latitude
  longitude
  recruits {
    ...RecruitFragment
  }
}
```

```
recruitsAggregate {  
  ...RecruitAggregateFragment  
}  
state  
}  
hometownId  
id  
jersey  
lastName  
name  
position {  
  abbreviation  
  athletes {  
    ...AthleteFragment  
  }  
  athletesAggregate {  
    ...AthleteAggregateFragment  
  }  
  displayName  
  id  
  name  
}  
positionId  
recruits {  
  athlete {  
    ...AthleteFragment  
  }  
  college {  
    ...currentTeamsFragment  
  }  
  height  
  hometown {  
    ...HometownFragment  
  }  
  id  
  name  
  overallRank  
  position {  
    ...RecruitPositionFragment  
  }  
}
```

```
        positionRank
        ranking
        rating
        recruitSchool {
          ...RecruitSchoolFragment
        }
        recruitType
        stars
        weight
        year
      }
      recruitsAggregate {
        aggregate {
          ...RecruitAggregateFieldsFr
        }
        nodes {
          ...RecruitFragment
        }
      }
      teamId
      weight
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 987,
  "offset": 123,
  "orderBy": [AthleteOrderBy],
  "where": AthleteBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "athlete": [  
      {  
        "adjustedPlayerMetrics": [A  
        "adjustedPlayerMetricsAggre  
        "athleteTeams": [AthleteTea  
        "athleteTeamsAggregate": At  
        "firstName": "xyz789",  
        "height": smallint,  
        "hometown": Hometown,  
        "hometownId": 123,  
        "id": bigint,  
        "jersey": smallint,  
        "lastName": "xyz789",  
        "name": "xyz789",  
        "position": Position,  
        "positionId": smallint,  
        "recruits": [Recruit],  
        "recruitsAggregate": Recrui  
        "teamId": 123,  
        "weight": smallint  
      }  
    ]  
  }  
}
```

Queries

athleteAggregate

fetch aggregated fields from the table:
"athlete"

Response

Returns an [AthleteAggregate!](#)

Arguments

Name	Description
distinctOn - [AthleteSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AthleteOrderBy!]	sort the rows by one or more columns
where - AthleteBoolExp	filter the rows returned

QUERY

```
query AthleteAggregate(
  $distinctOn: [AthleteSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteOrderBy!],
  $where: AthleteBoolExp
) {
  athleteAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...AthleteAvgFieldsFragment
      }
      count
      max {
        ...AthleteMaxFieldsFragment
      }
      min {
        ...AthleteMinFieldsFragment
      }
      stddev {
        ...AthleteStddevFieldsFragm
      }
      stddevPop {
        ...AthleteStddevPopFieldsFr
      }
      stddevSamp {
        ...AthleteStddevSampFieldsF
    
```

```
        }
      sum {
        ...AthleteSumFieldsFragment
      }
      varPop {
        ...AthleteVarPopFieldsFragment
      }
      varSamp {
        ...AthleteVarSampFieldsFragment
      }
      variance {
        ...AthleteVarianceFieldsFragment
      }
    }
  nodes {
    adjustedPlayerMetrics {
      ...AdjustedPlayerMetricsFragment
    }
    adjustedPlayerMetricsAggregate {
      ...AdjustedPlayerMetricsAggregate
    }
    athleteTeams {
      ...AthleteTeamFragment
    }
    athleteTeamsAggregate {
      ...AthleteTeamAggregateFragment
    }
    firstName
    height
    hometown {
      ...HometownFragment
    }
    hometownId
    id
    jersey
    lastName
    name
    position {
      ...PositionFragment
    }
  }
}
```

```
        positionId
        recruits {
          ...RecruitFragment
        }
        recruitsAggregate {
          ...RecruitAggregateFragment
        }
        teamId
        weight
      }
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 123,
  "offset": 987,
  "orderBy": [AthleteOrderBy],
  "where": AthleteBoolExp
}
```

RESPONSE

```
{
  "data": {
    "athleteAggregate": {
      "aggregate": AthleteAggregate
      "nodes": [Athlete]
    }
  }
}
```

Queries

athleteByPk

fetch data from the table: "athlete" using primary key columns

Response

Returns an [Athlete](#)

Arguments

Name	Description
<code>id - bigint!</code>	

QUERY

```
query AthleteByPk($id: bigint!) {
  athleteByPk(id: $id) {
    adjustedPlayerMetrics {
      athlete {
        ...AthleteFragment
      }
      athleteId
      metricType
      metricValue
      plays
      year
    }
    adjustedPlayerMetricsAggregate {
      aggregate {
        ...AdjustedPlayerMetricsAgg
      }
      nodes {
        ...AdjustedPlayerMetricsFra
      }
    }
    athleteTeams {
      ...
    }
  }
}
```

```
    athlete {
      ...AthleteFragment
    }
    athleteId
    endYear
    startYear
    team {
      ...historicalTeamFragment
    }
    teamId
  }
  athleteTeamsAggregate {
    aggregate {
      ...AthleteTeamAggregateField
    }
    nodes {
      ...AthleteTeamFragment
    }
  }
  firstName
  height
  hometown {
    athletes {
      ...AthleteFragment
    }
    athletesAggregate {
      ...AthleteAggregateFragment
    }
    city
    country
    countyFips
    latitude
    longitude
    recruits {
      ...RecruitFragment
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
    state
  }
```

```
        }
      hometownId
      id
      jersey
      lastName
      name
      position {
        abbreviation
        athletes {
          ...AthleteFragment
        }
      }
      athletesAggregate {
        ...AthleteAggregateFragment
      }
      displayName
      id
      name
    }
    positionId
    recruits {
      athlete {
        ...AthleteFragment
      }
      college {
        ...currentTeamsFragment
      }
      height
      hometown {
        ...HometownFragment
      }
      id
      name
      overallRank
      position {
        ...RecruitPositionFragment
      }
      positionRank
      ranking
      rating
      recruitSchool {
```

```
        ...RecruitSchoolFragment
    }
    recruitType
    stars
    weight
    year
}
recruitsAggregate {
    aggregate {
        ...RecruitAggregateFieldsFr
    }
    nodes {
        ...RecruitFragment
    }
}
teamId
weight
}
```

VARIABLES

```
{"id": bigint}
```

RESPONSE

```
{
  "data": {
    "athleteByPk": {
      "adjustedPlayerMetrics": [Adj
      "adjustedPlayerMetricsAggrega
      "athleteTeams": [AthleteTeam]
      "athleteTeamsAggregate": Athl
      "firstName": "xyz789",
      "height": smallint,
      "hometown": Hometown,
```

```
"hometownId": 987,  
"id": bigint,  
"jersey": smallint,  
"lastName": "xyz789",  
"name": "abc123",  
"position": Position,  
"positionId": smallint,  
"recruits": [Recruit],  
"recruitsAggregate": RecruitA  
"teamId": 987,  
"weight": smallint
```

Queries

athleteTeam

fetch data from the table:

"athlete_team"

Response

QUERY

Returns [AthleteTeam!]!

Arguments

```
query AthleteTeam(  
    $distinctOn: [AthleteTeamSelectCo  
    $limit: Int,  
    $offset: Int,  
    $orderBy: [AthleteTeamOrderBy!],
```

Name	Description
distinctOn - [AthleteTeamSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AthleteTeamOrderBy!]	sort the rows by one or more columns
where - AthleteTeamBoolExp	filter the rows returned

```
$where: AthleteTeamBoolExp
) {
  athleteTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
      positionId
      recruits {
        ...RecruitFragment
      }
      recruitsAggregate {
        ...
      }
    }
  }
}
```

```
    ...RecruitAggregateFragment
  }
  teamId
  weight
}
athleteId
endYear
startYear
team {
  abbreviation
  active
  altColor
  altName
  classification
  color
  conference
  conferenceAbbreviation
  conferenceId
  conferenceShortName
  countryCode
  displayName
  division
  endYear
  id
  images
  mascot
  ncaaName
  nickname
  school
  shortDisplayName
  startYear
  twitter
}
teamId
}
```

VARIABLES

```
{  
  "distinctOn": ["athleteId"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [AthleteTeamOrderBy],  
  "where": AthleteTeamBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "athleteTeam": [  
      {  
        "athlete": Athlete,  
        "athleteId": bigint,  
        "endYear": smallint,  
        "startYear": smallint,  
        "team": historicalTeam,  
        "teamId": 987  
      }  
    ]  
  }  
}
```

Queries

athleteTeamAggregate

fetch aggregated fields from the table:
 "athlete_team"

Response

Returns an [AthleteTeamAggregate!](#)

Arguments

Name	Description
distinctOn - [AthleteTeamSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AthleteTeamOrderBy!]	sort the rows by one or more columns
where - AthleteTeamBoolExp	filter the rows returned

QUERY

```
query AthleteTeamAggregate(
  $distinctOn: [AthleteTeamSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteTeamOrderBy!]!,
  $where: AthleteTeamBoolExp
) {
  athleteTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...AthleteTeamAvgFieldsFrag
      }
      count
      max {
        ...AthleteTeamMaxFieldsFrag
      }
      min {
        ...AthleteTeamMinFieldsFrag
      }
      stddev {
        ...AthleteTeamStddevFieldsF
      }
      stddevPop {
        ...AthleteTeamStddevPopFiel
      }
      stddevSamp {
        ...AthleteTeamStddevSampFie
      }
    }
  }
}
```

```
        }
      sum {
        ...AthleteTeamSumFieldsFrag
      }
      varPop {
        ...AthleteTeamVarPopFieldsF
      }
      varSamp {
        ...AthleteTeamVarSampFields
      }
      variance {
        ...AthleteTeamVarianceField
      }
    }
  nodes {
    athlete {
      ...AthleteFragment
    }
    athleteId
    endYear
    startYear
    team {
      ...historicalTeamFragment
    }
    teamId
  }
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 123,
  "offset": 123,
  "orderBy": [AthleteTeamOrderBy],
```

```
"where": AthleteTeamBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "athleteTeamAggregate": {
      "aggregate": AthleteTeamAggre
      "nodes": [AthleteTeam]
    }
  }
}
```



Queries

calendar

fetch data from the table: "calendar"

Response

Returns `[Calendar!]!`

Arguments

QUERY

```
query Calendar(
  $distinctOn: [CalendarSelectColumn]
  $limit: Int,
  $offset: Int,
```

Name	Description
distinctOn - [CalendarSelectColumns]	distinct select on [CalendarSelectColumns]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [CalendarOrderBy!]	sort the rows by one or more columns
where - CalendarBoolExp	filter the rows returned

```
$orderBy: [CalendarOrderBy!],
$where: CalendarBoolExp
) {
  calendar(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    endDate
    seasonType
    startDate
    week
    year
  }
}
```

VARIABLES

```
{
  "distinctOn": ["endDate"],
  "limit": 987,
  "offset": 123,
  "orderBy": [CalendarOrderBy!],
  "where": CalendarBoolExp
}
```

RESPONSE

```
{
  "data": {
    "calendar": [
      {
        "endDate": timestamp,
        "seasonType": season_type,
      }
    ]
  }
}
```

```

        "startDate": timestamp,
        "week": smallint,
        "year": smallint
    }
]
}
}

```

Queries

coach

fetch data from the table: "coach"

Response

Returns `[Coach!]!`

Arguments

Name	Description
<code>distinctOn - [CoachSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned

QUERY

```

query Coach(
  $distinctOn: [CoachSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachOrderBy!]!,
  $where: CoachBoolExp
) {
  coach(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  )
}

```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[CoachOrderBy!]</code>	sort the rows by one or more columns
where - <code>CoachBoolExp</code>	filter the rows returned

```

    ) {
      firstName
      id
      lastName
      seasons {
        coach {
          ...CoachFragment
        }
        games
        losses
        postseasonRank
        preseasonRank
        team {
          ...currentTeamsFragment
        }
        ties
        wins
        year
      }
      seasonsAggregate {
        aggregate {
          ...CoachSeasonAggregateField
        }
        nodes {
          ...CoachSeasonFragment
        }
      }
    }
  }
}

```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 123,
  "offset": 123,
  "orderBy": [CoachOrderBy],
```

```
        "where": CoachBoolExp
    }
```

RESPONSE

```
{
  "data": {
    "coach": [
      {
        "firstName": "abc123",
        "id": 123,
        "lastName": "xyz789",
        "seasons": [CoachSeason],
        "seasonsAggregate": CoachSe
      }
    ]
  }
}
```

Queries

coachAggregate

fetch aggregated fields from the table:
"coach"

Response

QUERY

Returns a [CoachAggregate!](#)

Arguments

Name	Description
distinctOn - [CoachSelectColumn!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [CoachOrderBy!]	sort the rows by one or more columns
where - CoachBoolExp	filter the rows returned

```
query CoachAggregate(
  $distinctOn: [CoachSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachOrderBy!]!,
  $where: CoachBoolExp
) {
  coachAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...CoachAvgFieldsFragment
      }
      count
      max {
        ...CoachMaxFieldsFragment
      }
      min {
        ...CoachMinFieldsFragment
      }
      stddev {
        ...CoachStddevFieldsFragmen
      }
      stddevPop {
        ...CoachStddevPopFieldsFrag
      }
      stddevSamp {
        ...CoachStddevSampFieldsFra
      }
      sum {
        ...CoachSumFieldsFragment
      }
      varPop {
        ...CoachVarPopFieldsFragmen
      }
    }
  }
}
```

```

    }
    varSamp {
      ...CoachVarSampFieldsFragme
    }
    variance {
      ...CoachVarianceFieldsFragm
    }
  }
  nodes {
    firstName
    id
    lastName
    seasons {
      ...CoachSeasonFragment
    }
    seasonsAggregate {
      ...CoachSeasonAggregateFrag
    }
  }
}

```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 987,
  "offset": 987,
  "orderBy": [CoachOrderBy],
  "where": CoachBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```

    "coachAggregate": {
      "aggregate": CoachAggregateFi
      "nodes": [Coach]
    }
  }
}

```

Queries

coachSeason

fetch data from the table:

"coach_season"

Response

Returns [CoachSeason!]!

Arguments

Name	Description
distinctOn - [CoachSeasonSelectColumns]	distinct select on
limit - Int	limit the number of rows returned

QUERY

```

query CoachSeason(
  $distinctOn: [CoachSeasonSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachSeasonOrderBy!],
  $where: CoachSeasonBoolExp
) {
  coachSeason(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
  )
}

```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[CoachSeasonOrderBy!]</code>	sort the rows by one or more columns
where - <code>CoachSeasonBoolExp</code>	filter the rows returned

```

  where: $where
} {
  coach {
    firstName
    id
    lastName
    seasons {
      ...CoachSeasonFragment
    }
    seasonsAggregate {
      ...CoachSeasonAggregateFrag
    }
  }
  games
  losses
  postseasonRank
  preseasonRank
  team {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
  ties
  wins
  year
}
}

```

VARIABLES

```
{
  "distinctOn": ["games"],
  "limit": 987,
  "offset": 123,
```

```
        "orderBy": [CoachSeasonOrderBy],  
        "where": CoachSeasonBoolExp  
    }  
}
```

RESPONSE

```
{  
  "data": {  
    "coachSeason": [  
      {  
        "coach": Coach,  
        "games": smallint,  
        "losses": smallint,  
        "postseasonRank": smallint,  
        "preseasonRank": smallint,  
        "team": currentTeams,  
        "ties": smallint,  
        "wins": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}
```



Queries

coachSeasonAggregate

fetch aggregated fields from the table:
 "coach_season"

Response

Returns a [CoachSeasonAggregate!](#)

Arguments

Name	Description
distinctOn - [CoachSeasonSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [CoachSeasonOrderBy!]	sort the rows by one or more columns
where - CoachSeasonBoolExp	filter the rows returned

QUERY

```
query CoachSeasonAggregate(
  $distinctOn: [CoachSeasonSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachSeasonOrderBy!],
  $where: CoachSeasonBoolExp
) {
  coachSeasonAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...CoachSeasonAvgFieldsFrag
      }
      count
      max {
        ...CoachSeasonMaxFieldsFrag
      }
      min {
        ...CoachSeasonMinFieldsFrag
      }
      stddev {
        ...CoachSeasonStddevFieldsF
      }
      stdDevPop {
        ...CoachSeasonStddevPopFiel
      }
      stdDevSamp {
        ...CoachSeasonStddevSampFie
    
```

```
        }
      sum {
        ...CoachSeasonSumFieldsFrag
      }
      varPop {
        ...CoachSeasonVarPopFieldsF
      }
      varSamp {
        ...CoachSeasonVarSampFields
      }
      variance {
        ...CoachSeasonVarianceField
      }
    }
  nodes {
    coach {
      ...CoachFragment
    }
    games
    losses
    postseasonRank
    preseasonRank
    team {
      ...currentTeamsFragment
    }
    ties
    wins
    year
  }
}
```

VARIABLES

```
{
  "distinctOn": ["games"],
  "limit": 987,
  "offset": 987,
```

```
"orderBy": [CoachSeasonOrderBy],  
"where": CoachSeasonBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "coachSeasonAggregate": {  
      "aggregate": CoachSeasonAggre  
      "nodes": [CoachSeason]  
    }  
  }  
}
```

Queries

conference

fetch data from the table: "conference"

Response

Returns [Conference!]!

QUERY

```
query Conference(  
  $distinctOn: [ConferenceSelectCol
```

Arguments

Name	Description
distinctOn - [ConferenceSelectColumn]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [ConferenceOrderBy!]	sort the rows by one or more columns
where - ConferenceBoolExp	filter the rows

```

$limit: Int,
$offset: Int,
$orderBy: [ConferenceOrderBy!],
$where: ConferenceBoolExp
) {
  conference(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    division
    id
    name
    shortName
    srName
  }
}

```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 987,
  "orderBy": [ConferenceOrderBy!],
  "where": ConferenceBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```

    "conference": [
      {
        "abbreviation": "abc123",
        "division": division,
        "id": smallint,
        "name": "xyz789",
        "shortName": "xyz789",
        "srName": "abc123"
      }
    ]
  }
}

```

Queries

currentTeams

fetch data from the table:

"current_conferences"

Response

Returns `[currentTeams!]!`

Arguments

QUERY

```

query CurrentTeams(
  $distinctOn: [currentTeamsSelectC
  $limit: Int,
  $offset: Int,
  $orderBy: [currentTeamsOrderBy!],

```

Name	Description
distinctOn - [currentTeamsSelectColumns!]	distinct select on [currentTeamsSelectColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [currentTeamsOrderBy!]	sort the rows by one or more columns
where - currentTeamsBoolExp	filter the rows returned

```
$where: currentTeamsBoolExp
) {
  currentTeams(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 987,
  "orderBy": [currentTeamsOrderBy],
  "where": currentTeamsBoolExp
}
```

RESPONSE

```
{
  "data": {
    "currentTeams": [
      {
        "id": 1234567890,
        "name": "Team A",
        "abbreviation": "TA",
        "classification": "Division I-A",
        "conference": "Big Ten Conference",
        "conferenceId": 1,
        "division": "Division I-A"
      }
    ]
  }
}
```

```
        "abbreviation": "abc123",
        "classification": division,
        "conference": "xyz789",
        "conferenceId": smallint,
        "division": "xyz789",
        "school": "xyz789",
        "teamId": 123
    }
]
}
}
```

Queries

currentTeamsAggregate

fetch aggregated fields from the table:
"current_conferences"

Response

Returns a [currentTeamsAggregate!](#)

Arguments

QUERY

```
query CurrentTeamsAggregate(
    $distinctOn: [currentTeamsSelectC
    $limit: Int,
    $offset: Int,
    $orderBy: [currentTeamsOrderBy!],
    $where: currentTeamsBoolExp
) {
```

Name	Description
distinctOn - [currentTeamsSelectColumns!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [currentTeamsOrderBy!]	sort the rows by one or more columns
where - currentTeamsBoolExpr!	filter the rows returned

```
currentTeamsAggregate(
  distinctOn: $distinctOn,
  limit: $limit,
  offset: $offset,
  orderBy: $orderBy,
  where: $where
) {
  aggregate {
    avg {
      ...currentTeamsAvgFieldsFra
    }
    count
    max {
      ...currentTeamsMaxFieldsFra
    }
    min {
      ...currentTeamsMinFieldsFra
    }
    stddev {
      ...currentTeamsStddevFields
    }
    stddevPop {
      ...currentTeamsStddevPopFie
    }
    stddevSamp {
      ...currentTeamsStddevSampFi
    }
    sum {
      ...currentTeamsSumFieldsFra
    }
    varPop {
      ...currentTeamsVarPopFields
    }
    varSamp {
      ...currentTeamsVarSampField
    }
    variance {
      ...currentTeamsVarianceFiel
    }
  }
}
```

```
    nodes {  
      abbreviation  
      classification  
      conference  
      conferenceId  
      division  
      school  
      teamId  
    }  
  }  
}
```

VARIABLES

```
{  
  "distinctOn": ["abbreviation"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [currentTeamsOrderBy],  
  "where": currentTeamsBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "currentTeamsAggregate": {  
      "aggregate": currentTeamsAggr  
      "nodes": [currentTeams]  
    }  
  }  
}
```

 Queries

draftPicks

fetch data from the table: "draft_picks"

Response

Returns `[DraftPicks!]!`

Arguments

Name	Description
<code>distinctOn - [DraftPicksSelectColumn!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [DraftPicksOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
query DraftPicks(
  $distinctOn: [DraftPicksSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [DraftPicksOrderBy!],
  $where: DraftPicksBoolExp
) {
  draftPicks(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    collegeAthleteRecord {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFra
      }
      adjustedPlayerMetricsAggregat
        ...AdjustedPlayerMetricsAgg
    }
    athleteTeams {
  
```

Name	Description
where - <code>DraftPicksBoolExp</code>	filter the rows returned

```

...AthleteTeamFragment
}
athleteTeamsAggregate {
  ...AthleteTeamAggregateFragment
}
firstName
height
hometown {
  ...HometownFragment
}
hometownId
id
jersey
lastName
name
position {
  ...PositionFragment
}
positionId
recruits {
  ...RecruitFragment
}
recruitsAggregate {
  ...RecruitAggregateFragment
}
teamId
weight
}
collegeId
collegeTeam {
  abbreviation
  active
  altColor
  altName
  classification
  color
  conference
  conferenceAbbreviation
  conferenceId
  conferenceShortName
}

```

```
countryCode
displayName
division
endYear
id
images
mascot
ncaaName
nickname
school
shortDisplayName
startYear
twitter
}
collegeTeamId
draftTeam {
  displayName
  id
  location
  logo
  mascot
  nickname
  picks {
    ...DraftPicksFragment
  }
  picksAggregate {
    ...DraftPicksAggregateFragm
  }
  shortDisplayName
}
grade
height
name
nflTeamId
overall
overallRank
pick
position {
  abbreviation
  id
}
```

```
        name
    }
    positionId
    positionRank
    round
    weight
    year
}
}
```

VARIABLES

```
{
  "distinctOn": ["collegeId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [DraftPicksOrderBy],
  "where": DraftPicksBoolExp
}
```

RESPONSE

```
{
  "data": {
    "draftPicks": [
      {
        "collegeAthleteRecord": Ath
        "collegeId": 123,
        "collegeTeam": historicalTe
        "collegeTeamId": 987,
        "draftTeam": DraftTeam,
        "grade": smallint,
        "height": smallint,
        "name": "abc123",
        "nflTeamId": smallint,
    
```

```
        "overall": smallint,
        "overallRank": smallint,
        "pick": smallint,
        "position": DraftPosition,
        "positionId": smallint,
        "positionRank": smallint,
        "round": smallint,
        "weight": smallint,
        "year": smallint
    }
]
}
}
```

Queries

draftPosition

fetch data from the table:

"draft_position"

Response

Returns [DraftPosition!]!

Arguments

QUERY

```
query DraftPosition(
  $distinctOn: [DraftPositionSelect
  $limit: Int,
  $offset: Int,
  $orderBy: [DraftPositionOrderBy!]]
```

Name	Description
distinctOn - [DraftPositionSelectColumns!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [DraftPositionOrderBy!]	sort the rows by one or more columns
where - DraftPositionBoolExp	filter the rows

```
$where: DraftPositionBoolExp
) {
  draftPosition(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    id
    name
  }
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 123,
  "offset": 123,
  "orderBy": [DraftPositionOrderBy]
  "where": DraftPositionBoolExp
}
```

RESPONSE

```
{
  "data": {
    "draftPosition": [
      {
        "abbreviation": "abc123",
        "id": smallint,
        "name": "abc123"
      }
    ]
  }
}
```

```
    ]
}
}
```

Queries

draftTeam

fetch data from the table: "draft_team"

Response

Returns `[DraftTeam!]!`

Arguments

Name	Description
<code>distinctOn - [DraftTeamSelectColumn]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with

QUERY

```
query DraftTeam(
  $distinctOn: [DraftTeamSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [DraftTeamOrderBy!],
  $where: DraftTeamBoolExp
) {
  draftTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    displayName
    id
  }
}
```

Name	Description
	order_by
orderBy - [DraftTeamOrderBy!]	sort the rows by one or more columns
where - DraftTeamBoolExp	filter the rows returned

```

location
logo
mascot
nickname
picks {
  collegeAthleteRecord {
    ...AthleteFragment
  }
  collegeId
  collegeTeam {
    ...historicalTeamFragment
  }
  collegeTeamId
  draftTeam {
    ...DraftTeamFragment
  }
  grade
  height
  name
  nflTeamId
  overall
  overallRank
  pick
  position {
    ...DraftPositionFragment
  }
  positionId
  positionRank
  round
  weight
  year
}
picksAggregate {
  aggregate {
    ...DraftPicksAggregateField
  }
  nodes {
    ...DraftPicksFragment
  }
}

```

```
        shortDisplayName  
    }  
}  
}
```

VARIABLES

```
{  
  "distinctOn": ["displayName"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [DraftTeamOrderBy],  
  "where": DraftTeamBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "draftTeam": [  
      {  
        "displayName": "xyz789",  
        "id": smallint,  
        "location": "abc123",  
        "logo": "xyz789",  
        "mascot": "abc123",  
        "nickname": "abc123",  
        "picks": [DraftPicks],  
        "picksAggregate": DraftPick  
        "shortDisplayName": "xyz789"  
      }  
    ]  
  }  
}
```

Queries

game

fetch data from the table: "game_info"

Response

Returns `[game!]!`

Arguments

Name	Description
<code>distinctOn - [gameSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [gameOrderBy!]</code>	sort the rows by one or more columns
<code>where - gameBoolExp</code>	filter the rows returned

QUERY

```
query Game(
  $distinctOn: [gameSelectColumn!]!,
  $limit: Int,
  $offset: Int,
  $orderBy: [gameOrderBy!],
  $where: gameBoolExp
) {
  game(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    attendance
    awayClassification
    awayConference
    awayConferenceId
    awayConferenceInfo {
      abbreviation
      division
      id
      name
      shortName
      srName
    }
  }
}
```

```
awayEndElo
awayLineScores
awayPoints
awayPostgameWinProb
awayStartElo
awayTeam
awayTeamId
awayTeamInfo {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
}
conferenceGame
excitement
homeClassification
homeConference
homeConferenceId
homeConferenceInfo {
    abbreviation
    division
    id
    name
    shortName
    srName
}
homeEndElo
homeLineScores
homePoints
homePostgameWinProb
homeStartElo
homeTeam
homeTeamId
homeTeamInfo {
    abbreviation
    classification
    conference
```

```
conferenceId
division
school
teamId
}
id
lines {
  gameId
  linesProviderId
  moneylineAway
  moneylineHome
  overUnder
  overUnderOpen
  provider {
    ...LinesProviderFragment
  }
  spread
  spreadOpen
}
linesAggregate {
  aggregate {
    ...GameLinesAggregateFields
  }
  nodes {
    ...GameLinesFragment
  }
}
mediaInfo {
  mediaType
  name
}
neutralSite
notes
season
seasonType
startDate
startTimeTbd
status
venueId
weather {
```

```
        condition {  
          ...WeatherConditionFragment  
        }  
        dewpoint  
        gameId  
        humidity  
        precipitation  
        pressure  
        snowfall  
        temperature  
        weatherConditionCode  
        windDirection  
        windGust  
        windSpeed  
      }  
    week  
  }  
}
```

VARIABLES

```
{  
  "distinctOn": ["attendance"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [gameOrderBy],  
  "where": gameBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "game": [  
      {  
        "attendance": 987,  
        "date": "2023-01-01T10:00:00Z",  
        "division": "West",  
        "gameId": "2023-01-01T10:00:00Z",  
        "homeTeam": "University of Texas",  
        "homeTeamAbbr": "UT",  
        "homeTeamLogo": "https://cfbd.com/assets/images/team-logos/ut.png",  
        "homeTeamScore": 24,  
        "isHomeGame": true,  
        "isNeutralSite": false,  
        "isNightGame": false,  
        "isOvertime": false,  
        "isRegularSeason": true,  
        "isSundayGame": false,  
        "isWeekendGame": false,  
        "isWinterGame": false,  
        "opponent": "University of Oklahoma",  
        "opponentAbbr": "OU",  
        "opponentLogo": "https://cfbd.com/assets/images/team-logos/ou.png",  
        "opponentScore": 21,  
        "quarter": 1,  
        "quarterTime": "10:00 AM",  
        "quarterOrder": 1,  
        "quarterType": "Normal",  
        "score": 24,  
        "status": "Completed",  
        "time": "10:00 AM",  
        "timeOffset": "00:00:00",  
        "timeZone": "EST",  
        "week": 1  
      }  
    ]  
  }  
}
```

```
"awayClassification": divis
"awayConference": "abc123",
"awayConferenceId": smallint
"awayConferenceInfo": ConferenceInfo
"awayEndElo": 123,
"awayLineScores": [smallint]
"awayPoints": smallint,
"awayPostgameWinProb": numeric
"awayStartElo": 123,
"awayTeam": "xyz789",
"awayTeamId": 123,
"awayTeamInfo": currentTeam
"conferenceGame": true,
"excitement": numeric,
"homeClassification": divis
"homeConference": "abc123",
"homeConferenceId": smallint
"homeConferenceInfo": ConferenceInfo
"homeEndElo": 987,
"homeLineScores": [smallint]
"homePoints": smallint,
"homePostgameWinProb": numeric
"homeStartElo": 987,
"homeTeam": "abc123",
"homeTeamId": 123,
"homeTeamInfo": currentTeam
"id": 987,
"lines": [GameLines],
"linesAggregate": GameLines
"mediaInfo": [GameMedia],
"neutralSite": true,
"notes": "abc123",
"season": smallint,
"seasonType": season_type,
"startDate": timestamp,
"startTimeTbd": true,
"status": game_status,
"venueId": 987,
"weather": GameWeather,
"week": smallint
```

```

        }
    ]
}
}
```

Queries

gameAggregate

fetch aggregated fields from the table:

"game_info"

Response

Returns a [gameAggregate!](#)

Arguments

Name	Description
<code>distinctOn - [gameSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned

QUERY

```

query GameAggregate(
  $distinctOn: [gameSelectColumn!]!,
  $limit: Int,
  $offset: Int,
  $orderBy: [gameOrderBy!]!,
  $where: gameBoolExp
) {
  gameAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[gameOrderBy!]</code>	sort the rows by one or more columns
where - <code>gameBoolExp</code>	filter the rows returned

```

avg {
  ...gameAvgFieldsFragment
}
count
max {
  ...gameMaxFieldsFragment
}
min {
  ...gameMinFieldsFragment
}
stddev {
  ...gameStddevFieldsFragment
}
stddevPop {
  ...gameStddevPopFieldsFragm
}
stddevSamp {
  ...gameStddevSampFieldsFrag
}
sum {
  ...gameSumFieldsFragment
}
varPop {
  ...gameVarPopFieldsFragment
}
varSamp {
  ...gameVarSampFieldsFragmen
}
variance {
  ...gameVarianceFieldsFragme
}
}
nodes {
  attendance
  awayClassification
  awayConference
  awayConferenceId
  awayConferenceInfo {
    ...ConferenceFragment
}

```

```
awayEndElo
awayLineScores
awayPoints
awayPostgameWinProb
awayStartElo
awayTeam
awayTeamId
awayTeamInfo {
    ...currentTeamsFragment
}
conferenceGame
excitement
homeClassification
homeConference
homeConferenceId
homeConferenceInfo {
    ...ConferenceFragment
}
homeEndElo
homeLineScores
homePoints
homePostgameWinProb
homeStartElo
homeTeam
homeTeamId
homeTeamInfo {
    ...currentTeamsFragment
}
id
lines {
    ...GameLinesFragment
}
linesAggregate {
    ...GameLinesAggregateFragme
}
mediaInfo {
    ...GameMediaFragment
}
neutralSite
notes
```

```
    season
    seasonType
    startDate
    startTimeTbd
    status
    venueId
    weather {
        ...GameWeatherFragment
    }
    week
}
}
```

VARIABLES

```
{
    "distinctOn": ["attendance"],
    "limit": 123,
    "offset": 123,
    "orderBy": [gameOrderBy],
    "where": gameBoolExp
}
```

RESPONSE

```
{
    "data": {
        "gameAggregate": {
            "aggregate": gameAggregateFie
            "nodes": [game]
        }
    }
}
```

 Queries

gameLines

fetch data from the table: "game_lines"

Response

Returns `[GameLines!]!`

Arguments

Name	Description
<code>distinctOn - [GameLinesSelectColumns]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [GameLinesOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
query GameLines(
  $distinctOn: [GameLinesSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [GameLinesOrderBy!],
  $where: GameLinesBoolExp
) {
  gameLines(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    gameId
    linesProviderId
    moneylineAway
    moneylineHome
    overUnder
    overUnderOpen
    provider {
      id
    }
  }
}
```

Name	Description	
where - <code>GameLinesBoolExp</code>	filter the rows returned	<pre> name } spread spreadOpen } }</pre>

VARIABLES

```
{
  "distinctOn": ["gameId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [GameLinesOrderBy],
  "where": GameLinesBoolExp
}
```

RESPONSE

```
{
  "data": {
    "gameLines": [
      {
        "gameId": 987,
        "linesProviderId": 123,
        "moneylineAway": 123,
        "moneylineHome": 987,
        "overUnder": numeric,
        "overUnderOpen": numeric,
        "provider": LinesProvider,
        "spread": numeric,
        "spreadOpen": numeric
      }
    ]
}
```

```
}
```

Queries

gameLinesAggregate

fetch aggregated fields from the table:
 "game_lines"

Response

Returns a [GameLinesAggregate!](#)

Arguments

Name	Description
distinctOn - [GameLinesSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

QUERY

```
query GameLinesAggregate(
  $distinctOn: [GameLinesSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [GameLinesOrderBy!],
  $where: GameLinesBoolExp
) {
  gameLinesAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...GameLinesAvgFieldsFragme
    }
  }
}
```

Name	Description
orderBy - [GameLinesOrderBy!]	sort the rows by one or more columns
where - GameLinesBoolExp	filter the rows returned

```

    }
    count
    max {
      ...GameLinesMaxFieldsFragme
    }
    min {
      ...GameLinesMinFieldsFragme
    }
    stdDev {
      ...GameLinesStddevFieldsFra
    }
    stdDevPop {
      ...GameLinesStddevPopFields
    }
    stdDevSamp {
      ...GameLinesStddevSampField
    }
    sum {
      ...GameLinesSumFieldsFragme
    }
    varPop {
      ...GameLinesVarPopFieldsFra
    }
    varSamp {
      ...GameLinesVarSampFieldsFr
    }
    variance {
      ...GameLinesVarianceFieldsF
    }
  }
  nodes {
    gameId
    linesProviderId
    moneylineAway
    moneylineHome
    overUnder
    overUnderOpen
    provider {
      ...LinesProviderFragment
    }
  }
}

```

```
        spread
        spreadOpen
      }
    }
}
```

VARIABLES

```
{
  "distinctOn": ["gameId"],
  "limit": 987,
  "offset": 123,
  "orderBy": [GameLinesOrderBy],
  "where": GameLinesBoolExp
}
```

RESPONSE

```
{
  "data": {
    "gameLinesAggregate": {
      "aggregate": GameLinesAggregate,
      "nodes": [GameLines]
    }
  }
}
```



Queries

gameMedia

fetch data from the table: "game_media"

Response

Returns `[GameMedia!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[GameMediaSelectColumn!]</code>	distinct select on columns
<code>limit</code> - <code>Int</code>	limit the number of rows returned
<code>offset</code> - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy</code> - <code>[GameMediaOrderBy!]</code>	sort the rows by one or more columns
<code>where</code> - <code>GameMediaBoolExp</code>	filter the rows returned

QUERY

```
query GameMedia(
  $distinctOn: [GameMediaSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [GameMediaOrderBy!],
  $where: GameMediaBoolExp
) {
  gameMedia(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    mediaType
    name
  }
}
```

VARIABLES

```
{
  "distinctOn": ["mediaType"],
  "limit": 987,
  "offset": 123,
  "orderBy": [GameMediaOrderBy],
```

```

    "where": GameMediaBoolExp
}

```

RESPONSE

```
{
  "data": {
    "gameMedia": [
      {
        "mediaType": media_type,
        "name": "xyz789"
      }
    ]
  }
}
```

Queries**gamePlayerStat**

fetch data from the table:

"game_player_stat"

Response

Returns **[GamePlayerStat!]!**

Arguments**QUERY**

```

query GamePlayerStat(
  $distinctOn: [GamePlayerStatSelect]
  $limit: Int,
  $offset: Int,
)

```

Name	Description
distinctOn - [GamePlayerStatSelectableColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GamePlayerStatOrderBy!]	sort the rows by one or more columns
where - GamePlayerStatBoolExp	filter the rows

```
$orderBy: [GamePlayerStatOrderBy!]!
$where: GamePlayerStatBoolExp
) {
  gamePlayerStat(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
      positionId
      recruits {
        ...RecruitFragment
      }
    }
  }
}
```

```
recruitsAggregate {  
  ...RecruitAggregateFragment  
}  
teamId  
weight  
}  
athleteId  
gameTeam {  
  endElo  
  game {  
    ...gameFragment  
  }  
  gameId  
  gamePlayerStats {  
    ...GamePlayerStatFragment  
  }  
  gamePlayerStatsAggregate {  
    ...GamePlayerStatAggregateF  
  }  
  homeAway  
  lineScores  
  points  
  startElo  
  teamId  
  winProb  
}  
gameTeamId  
id  
playerStatCategory {  
  gamePlayerStats {  
    ...GamePlayerStatFragment  
  }  
  gamePlayerStatsAggregate {  
    ...GamePlayerStatAggregateF  
  }  
  name  
}  
playerStatType {  
  name  
}
```

```
    stat
  }
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [GamePlayerStatOrderBy],
  "where": GamePlayerStatBoolExp
}
```



RESPONSE

```
{
  "data": {
    "gamePlayerStat": [
      {
        "athlete": Athlete,
        "athleteId": bigint,
        "gameTeam": GameTeam,
        "gameTeamId": bigint,
        "id": bigint,
        "playerStatCategory": PlayerStatCategory,
        "playerStatType": PlayerStatType,
        "stat": "xyz789"
      }
    ]
  }
}
```



Queries

gamePlayerStatAggregate

fetch aggregated fields from the table:

"game_player_stat"

Response

Returns a [GamePlayerStatAggregate!](#)

Arguments

Name	Description
distinctOn - [GamePlayerStatSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GamePlayerStatOrderBy!]	sort the rows by

QUERY

```
query GamePlayerStatAggregate(
  $distinctOn: [GamePlayerStatSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [GamePlayerStatOrderBy!]!
  $where: GamePlayerStatBoolExp
) {
  gamePlayerStatAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...GamePlayerStatAvgFieldsF
      }
      count
      max {
        ...GamePlayerStatMaxFieldsF
      }
    }
  }
}
```

Name	Description
	columns
where -	filter the rows
GamePlayerStatBool	returned

```

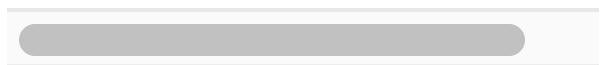
}
min {
  ...GamePlayerStatMinFieldsF
}
stddev {
  ...GamePlayerStatStddevFiel
}
stddevPop {
  ...GamePlayerStatStddevPopF
}
stddevSamp {
  ...GamePlayerStatStddevSamp
}
sum {
  ...GamePlayerStatSumFieldsF
}
varPop {
  ...GamePlayerStatVarPopFiel
}
varSamp {
  ...GamePlayerStatVarSampFie
}
variance {
  ...GamePlayerStatVarianceFi
}
}
nodes {
  athlete {
    ...AthleteFragment
  }
  athleteId
  gameTeam {
    ...GameTeamFragment
  }
  gameTeamId
  id
  playerStatCategory {
    ...PlayerStatCategoryFragme
  }
  playerStatType {

```

```
    ...PlayerStatTypeFragment
}
stat
}
}
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 123,
  "offset": 123,
  "orderBy": [GamePlayerStatOrderBy],
  "where": GamePlayerStatBoolExp
}
```



RESPONSE

```
{
  "data": {
    "gamePlayerStatAggregate": {
      "aggregate": GamePlayerStatAgg,
      "nodes": [GamePlayerStat]
    }
  }
}
```



Queries

gameTeam

fetch data from the table: "game_team"

Response

Returns `[GameTeam!]!`

Arguments

Name	Description
<code>distinctOn - [GameTeamSelectColumns]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [GameTeamOrderBy!]</code>	sort the rows by one or more columns
<code>where - GameTeamBoolExp</code>	filter the rows returned

QUERY

```
query GameTeam(
  $distinctOn: [GameTeamSelectColum
  $limit: Int,
  $offset: Int,
  $orderBy: [GameTeamOrderBy!],
  $where: GameTeamBoolExp
) {
  gameTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    endElo
    game {
      attendance
      awayClassification
      awayConference
      awayConferenceId
      awayConferenceInfo {
        ...ConferenceFragment
      }
      awayEndElo
      awayLineScores
      awayPoints
    }
  }
}
```

```
awayPostgameWinProb
awayStartElo
awayTeam
awayTeamId
awayTeamInfo {
    ...currentTeamsFragment
}
conferenceGame
excitement
homeClassification
homeConference
homeConferenceId
homeConferenceInfo {
    ...ConferenceFragment
}
homeEndElo
homeLineScores
homePoints
homePostgameWinProb
homeStartElo
homeTeam
homeTeamId
homeTeamInfo {
    ...currentTeamsFragment
}
id
lines {
    ...GameLinesFragment
}
linesAggregate {
    ...GameLinesAggregateFragme
}
mediaInfo {
    ...GameMediaFragment
}
neutralSite
notes
season
seasonType
startDate
```

```
startTimeTbd
status
venueId
weather {
    ...GameWeatherFragment
}
week
}
gameId
gamePlayerStats {
    athlete {
        ...AthleteFragment
    }
    athleteId
    gameTeam {
        ...GameTeamFragment
    }
    gameTeamId
    id
    playerStatCategory {
        ...PlayerStatCategoryFragme
    }
    playerStatType {
        ...PlayerStatTypeFragment
    }
    stat
}
gamePlayerStatsAggregate {
    aggregate {
        ...GamePlayerStatAggregateF
    }
    nodes {
        ...GamePlayerStatFragment
    }
}
homeAway
lineScores
points
startElo
teamId
```

```
        winProb
    }
}
```

VARIABLES

```
{
  "distinctOn": ["endElo"],
  "limit": 123,
  "offset": 123,
  "orderBy": [GameTeamOrderBy],
  "where": GameTeamBoolExp
}
```

RESPONSE

```
{
  "data": {
    "gameTeam": [
      {
        "endElo": 987,
        "game": game,
        "gameId": 987,
        "gamePlayerStats": [GamePla
        "gamePlayerStatsAggregate": [
          "homeAway": home_away,
          "lineScores": [smallint],
          "points": smallint,
          "startElo": 123,
          "teamId": 987,
          "winProb": numeric
        ]
      }
    ]
  }
}
```

```

    }
}
}
```

Queries

gameWeather

fetch data from the table:
"game_weather"

Response

Returns `[GameWeather!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[GameWeatherSelectColumns]</code>	distinct select on
<code>limit</code> - <code>Int</code>	limit the number of rows returned
<code>offset</code> - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>

QUERY

```

query GameWeather(
  $distinctOn: [GameWeatherSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [GameWeatherOrderBy!],
  $where: GameWeatherBoolExp
) {
  gameWeather(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    condition {
      description
      id
    }
  }
}
```

Name	Description
orderBy - [GameWeatherOrderBy!]	sort the rows by one or more columns
where - GameWeatherBoolExp	filter the rows returned

```

        }
        dewpoint
        gameId
        humidity
        precipitation
        pressure
        snowfall
        temperature
        weatherConditionCode
        windDirection
        windGust
        windSpeed
    }
}

```

VARIABLES

```
{
  "distinctOn": ["dewpoint"],
  "limit": 123,
  "offset": 987,
  "orderBy": [GameWeatherOrderBy],
  "where": GameWeatherBoolExp
}
```



RESPONSE

```
{
  "data": {
    "gameWeather": [
      {
        "condition": WeatherConditi
        "dewpoint": numeric,
        "gameId": 123,
        "humidity": numeric,

```

```
        "precipitation": numeric,  
        "pressure": numeric,  
        "snowfall": numeric,  
        "temperature": numeric,  
        "weatherConditionCode": sma  
        "windDirection": numeric,  
        "windGust": numeric,  
        "windSpeed": numeric  
    }  
}  
]  
}  
}
```

Queries

historicalTeam

fetch data from the table: "team_info"

Response

Returns [historicalTeam!]!

Arguments

QUERY

```
query HistoricalTeam(  
  $distinctOn: [historicalTeamSele  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [historicalTeamOrderBy!  
  $where: historicalTeamBoolExp  
) {
```

Name	Description
distinctOn - [historicalTeamSelection!]	distinct select on [historicalTeamSelection!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [historicalTeamOrderBy!]	sort the rows by one or more columns
where - historicalTeamBoolean	filter the rows

```

historicalTeam(
  distinctOn: $distinctOn,
  limit: $limit,
  offset: $offset,
  orderBy: $orderBy,
  where: $where
) {
  abbreviation
  active
  altColor
  altName
  classification
  color
  conference
  conferenceAbbreviation
  conferenceId
  conferenceShortName
  countryCode
  displayName
  division
  endYear
  id
  images
  mascot
  ncaaName
  nickname
  school
  shortDisplayName
  startYear
  twitter
}
}

```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 123,
```

```
        "offset": 123,  
        "orderBy": [historicalTeamOrderBy  
        "where": historicalTeamBoolExp  
    }]
```

RESPONSE

```
{  
  "data": {  
    "historicalTeam": [  
      {  
        "abbreviation": "abc123",  
        "active": true,  
        "altColor": "abc123",  
        "altName": "abc123",  
        "classification": division,  
        "color": "xyz789",  
        "conference": "abc123",  
        "conferenceAbbreviation": "  
        "conferenceId": smallint,  
        "conferenceShortName": "xyz",  
        "countryCode": "abc123",  
        "displayName": "abc123",  
        "division": "abc123",  
        "endYear": smallint,  
        "id": 123,  
        "images": ["abc123"],  
        "mascot": "xyz789",  
        "ncaaName": "abc123",  
        "nickname": "xyz789",  
        "school": "xyz789",  
        "shortDisplayName": "xyz789",  
        "startYear": smallint,  
        "twitter": "xyz789"  
      }  
    ]  
}
```

```

    }
}

```

Queries

historicalTeamAggregate

fetch aggregated fields from the table:

"team_info"

Response

Returns a [historicalTeamAggregate](#)!

Arguments

Name	Description
distinctOn - [historicalTeamSelectionColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

QUERY

```

query HistoricalTeamAggregate(
  $distinctOn: [historicalTeamSelectionColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [historicalTeamOrderBy!]!
  $where: historicalTeamBoolExp
) {
  historicalTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...historicalTeamAvgFieldsF
      }
    }
  }
}

```

Name	Description
orderBy - [historicalTeamOrderBy!]	sort the rows by one or more columns
where - historicalTeamBoolRef	filter the rows

```

}
count
max {
  ...historicalTeamMaxFieldsF
}
min {
  ...historicalTeamMinFieldsF
}
stddev {
  ...historicalTeamStddevFiel
}
stddevPop {
  ...historicalTeamStddevPopF
}
stddevSamp {
  ...historicalTeamStddevSamp
}
sum {
  ...historicalTeamSumFieldsF
}
varPop {
  ...historicalTeamVarPopFiel
}
varSamp {
  ...historicalTeamVarSampFie
}
variance {
  ...historicalTeamVarianceFi
}
}
nodes {
  abbreviation
  active
  altColor
  altName
  classification
  color
  conference
  conferenceAbbreviation
  conferenceId
}
```

```
conferenceShortName
countryCode
displayName
division
endYear
id
images
mascot
ncaaName
nickname
school
shortDisplayName
startYear
twitter
}
}
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 987,
  "orderBy": [historicalTeamOrderBy],
  "where": historicalTeamBoolExp
}
```

RESPONSE

```
{
  "data": {
    "historicalTeamAggregate": {
      "aggregate": historicalTeamAgg
      "nodes": [historicalTeam]
    }
  }
}
```

{
}
}
}

Queries

hometown

fetch data from the table: "hometown"

Response

Returns `[Hometown!]!`

Arguments

Name	Description
<code>distinctOn - [HometownSelectColumns]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with

QUERY

```
query Hometown(  
  $distinctOn: [HometownSelectColum  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [HometownOrderBy!],  
  $where: HometownBoolExp  
) {  
  hometown(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    athletes {  
      adjustedPlayerMetrics {
```

Name	Description
order_by	
orderBy - [HometownOrderBy!]	sort the rows by one or more columns
where - HometownBoolExp	filter the rows returned

```

...AdjustedPlayerMetricsFra
}
adjustedPlayerMetricsAggregat
...AdjustedPlayerMetricsAgg
}
athleteTeams {
...AthleteTeamFragment
}
athleteTeamsAggregate {
...AthleteTeamAggregateFrag
}
firstName
height
hometown {
...HometownFragment
}
hometownId
id
jersey
lastName
name
position {
...PositionFragment
}
positionId
recruits {
...RecruitFragment
}
recruitsAggregate {
...RecruitAggregateFragment
}
teamId
weight
}
athletesAggregate {
aggregate {
...AthleteAggregateFieldsFr
}
nodes {
...AthleteFragment
}

```

```
        }
    }
    city
    country
    countyFips
    latitude
    longitude
    recruits {
        athlete {
            ...AthleteFragment
        }
        college {
            ...currentTeamsFragment
        }
        height
        hometown {
            ...HometownFragment
        }
        id
        name
        overallRank
        position {
            ...RecruitPositionFragment
        }
        positionRank
        ranking
        rating
        recruitSchool {
            ...RecruitSchoolFragment
        }
        recruitType
        stars
        weight
        year
    }
    recruitsAggregate {
        aggregate {
            ...RecruitAggregateFieldsFr
        }
        nodes {
    }
```

```
        ...RecruitFragment
    }
}
state
}
```

VARIABLES

```
{
  "distinctOn": ["city"],
  "limit": 987,
  "offset": 987,
  "orderBy": [HometownOrderBy],
  "where": HometownBoolExp
}
```

RESPONSE

```
{
  "data": {
    "hometown": [
      {
        "athletes": [Athlete],
        "athletesAggregate": Athlet
        "city": "xyz789",
        "country": "abc123",
        "countyFips": "abc123",
        "latitude": numeric,
        "longitude": numeric,
        "recruits": [Recruit],
        "recruitsAggregate": Recrui
        "state": "xyz789"
      }
    ]
  }
}
```

```

    }
}

```

Queries

hometownAggregate

fetch aggregated fields from the table:
"hometown"

Response

Returns a [HometownAggregate!](#)

Arguments

Name	Description
distinctOn - [HometownSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

QUERY

```

query HometownAggregate(
  $distinctOn: [HometownSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [HometownOrderBy!]!,
  $where: HometownBoolExp
) {
  hometownAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...HometownAvgFieldsFragment
      }
    }
  }
}

```

Name	Description
orderBy - [HometownOrderBy!]	sort the rows by one or more columns
where - HometownBoolExp	filter the rows returned

```

    }
    count
    max {
      ...HometownMaxFieldsFragmen
    }
    min {
      ...HometownMinFieldsFragmen
    }
    stdDev {
      ...HometownStddevFieldsFrag
    }
    stdDevPop {
      ...HometownStddevPopFieldsF
    }
    stdDevSamp {
      ...HometownStddevSampFields
    }
    sum {
      ...HometownSumFieldsFragmen
    }
    varPop {
      ...HometownVarPopFieldsFrag
    }
    varSamp {
      ...HometownVarSampFieldsFra
    }
    variance {
      ...HometownVarianceFieldsFr
    }
  }
  nodes {
    athletes {
      ...AthleteFragment
    }
    athletesAggregate {
      ...AthleteAggregateFragment
    }
    city
    country
    countyFips
  }
}

```

```
        latitude
        longitude
      recruits {
        ...RecruitFragment
      }
      recruitsAggregate {
        ...RecruitAggregateFragment
      }
      state
    }
  }
}
```

VARIABLES

```
{
  "distinctOn": ["city"],
  "limit": 123,
  "offset": 123,
  "orderBy": [HometownOrderBy],
  "where": HometownBoolExp
}
```

RESPONSE

```
{
  "data": {
    "hometownAggregate": {
      "aggregate": HometownAggregate,
      "nodes": [Hometown]
    }
  }
}
```

Queries

linesProvider

fetch data from the table:

"lines_provider"

Response

Returns `[LinesProvider!]!`

Arguments

Name	Description
<code>distinctOn - [LinesProviderSelectColumns!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [LinesProviderOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
query LinesProvider(
  $distinctOn: [LinesProviderSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [LinesProviderOrderBy!]!
  $where: LinesProviderBoolExp
) {
  linesProvider(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    name
  }
}
```

VARIABLES

Name	Description
where - <code>LinesProviderBoolExp</code>	filter the rows

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [LinesProviderOrderBy]  
  "where": LinesProviderBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "linesProvider": [  
      {"id": 123, "name": "xyz789"}  
    ]  
  }  
}
```

Queries

linesProviderAggregate

fetch aggregated fields from the table:
"lines_provider"

Response

Returns a [LinesProviderAggregate!](#)

Arguments

Name	Description
distinctOn - [LinesProviderSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [LinesProviderOrderBy!]	sort the rows by one or more columns
where - LinesProviderBoolExp	filter the rows

QUERY

```

query LinesProviderAggregate(
  $distinctOn: [LinesProviderSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [LinesProviderOrderBy!]!
  $where: LinesProviderBoolExp
) {
  linesProviderAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...LinesProviderAvgFieldsFr
      }
      count
      max {
        ...LinesProviderMaxFieldsFr
      }
      min {
        ...LinesProviderMinFieldsFr
      }
      stdDev {
        ...LinesProviderStddevField
      }
      stdDevPop {
        ...LinesProviderStddevPopFi
      }
      stdDevSamp {
        ...LinesProviderStddevSampF
      }
      sum {
        ...LinesProviderSumFieldsFr
      }
    }
  }
}

```

```
    varPop {  
      ...LinesProviderVarPopField  
    }  
    varSamp {  
      ...LinesProviderVarSampFiel  
    }  
    variance {  
      ...LinesProviderVarianceFie  
    }  
  }  
  nodes {  
    id  
    name  
  }  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [LinesProviderOrderBy]  
  "where": LinesProviderBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "linesProviderAggregate": {  
      "aggregate": LinesProviderAgg  
      "nodes": [LinesProvider]  
    }  
  }  
}
```

```
}
```

Queries

playerStatCategory

fetch data from the table:

"player_stat_category"

Response

Returns [PlayerStatCategory!]!

Arguments

Name	Description
distinctOn - [PlayerStatCategoryColumns!]	distinct select on [PlayerStatCategoryColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with

QUERY

```
query PlayerStatCategory(  
  $distinctOn: [PlayerStatCategory$  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [PlayerStatCategoryOrder  
  $where: PlayerStatCategoryBoolExp  
) {  
  playerStatCategory(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    gamePlayerStats {  
      athlete {
```

Name	Description
	order_by
orderBy - [PlayerStatCategory] orderBy!	sort the rows by one or more columns
where - PlayerStatCategory	filter the rows Returned

```
    ...AthleteFragment
}
athleteId
gameTeam {
    ...GameTeamFragment
}
gameTeamId
id
playerStatCategory {
    ...PlayerStatCategoryFragme
}
playerStatType {
    ...PlayerStatTypeFragment
}
stat
}
gamePlayerStatsAggregate {
    aggregate {
        ...GamePlayerStatAggregateF
}
nodes {
    ...GamePlayerStatFragment
}
}
name
```

VARIABLES

```
{  
  "distinctOn": ["name"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [PlayerStatCategoryOrd
```

```
"where": PlayerStatCategoryBoolEx
```

```
}
```

RESPONSE

```
{
  "data": {
    "playerStatCategory": [
      {
        "gamePlayerStats": [GamePla
        "gamePlayerStatsAggregate":
        "name": "abc123"
      }
    ]
  }
}
```



Queries

playerStatCategoryAggregate

fetch aggregated fields from the table:
"player_stat_category"

Response

QUERY

Returns a
[PlayerStatCategoryAggregate!](#)

Arguments

Name	Description
distinctOn - [PlayerStatCategoryColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatCategoryOrderBy!]	sort the rows by one or more columns
where - PlayerStatCategoryReturned	filter the rows

```
query PlayerStatCategoryAggregate(
  $distinctOn: [PlayerStatCategoryColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PlayerStatCategoryOrder]
  $where: PlayerStatCategoryBoolExp
) {
  playerStatCategoryAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      count
      max {
        ...PlayerStatCategoryMaxField
      }
      min {
        ...PlayerStatCategoryMinField
      }
    }
    nodes {
      gamePlayerStats {
        ...GamePlayerStatFragment
      }
      gamePlayerStatsAggregate {
        ...GamePlayerStatAggregateField
      }
      name
    }
  }
}
```

VARIABLES

```
{  
  "distinctOn": ["name"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [PlayerStatCategoryOrdering],  
  "where": PlayerStatCategoryBoolExpr}
```

RESPONSE

```
{  
  "data": {  
    "playerStatCategoryAggregate": {  
      "aggregate": PlayerStatCategoryAggregate,  
      "nodes": [PlayerStatCategory]...  
    }  
  }  
}
```

Queries

playerStatType

fetch data from the table:

"player_stat_type"

Response**Returns [PlayerStatType!]!****Arguments**

Name	Description
distinctOn - [PlayerStatTypeSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatTypeOrderBy!]	sort the rows by one or more columns
where - PlayerStatTypeBoolExp	filter the rows

QUERY

```
query PlayerStatType(
  $distinctOn: [PlayerStatTypeSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PlayerStatTypeOrderBy!]!
  $where: PlayerStatTypeBoolExp
) {
  playerStatType(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    name
  }
}
```

VARIABLES

```
{
  "distinctOn": ["name"],
  "limit": 987,
  "offset": 987,
  "orderBy": [PlayerStatTypeOrderBy!],
  "where": PlayerStatTypeBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "playerStatType": [{"name": "ab"}]  
  }  
}
```

Queries

playerStatTypeAggregate

fetch aggregated fields from the table:
"player_stat_type"

Response

Returns a [PlayerStatTypeAggregate!](#)

Arguments

Name	Description
distinctOn - [PlayerStatTypeSelection!]	distinct select on [PlayerStatTypeSelection!]
limit - Int	limit the number of rows returned

QUERY

```
query PlayerStatTypeAggregate(  
  $distinctOn: [PlayerStatTypeSele  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [PlayerStatTypeOrderBy!  
  $where: PlayerStatTypeBoolExp  
) {  
  playerStatTypeAggregate(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[PlayerStatTypeOrderBy!]</code>	sort the rows by one or more columns
where - <code>PlayerStatTypeBoolExp</code>	filter the rows

```
) {
  aggregate: {
    count,
    max: ...PlayerStatTypeMaxFieldsF
  }
  min: ...PlayerStatTypeMinFieldsF
}
nodes: {
  name
}
}
```

VARIABLES

```
{
  "distinctOn": ["name"],
  "limit": 987,
  "offset": 123,
  "orderBy": [PlayerStatTypeOrderBy],
  "where": PlayerStatTypeBoolExp
}
```

RESPONSE

```
{
  "data": {
    "playerStatTypeAggregate": {
      "aggregate": PlayerStatTypeAg
      "nodes": [PlayerStatType]
    }
  }
}
```

```

        }
    }
}
```

Queries

poll

fetch data from the table: "poll"

Response

Returns `[Poll!]!`

Arguments

Name	Description
<code>distinctOn - [PollSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>

QUERY

```

query Poll(
  $distinctOn: [PollSelectColumn!],
  $limit: Int,
  $offset: Int,
  $orderBy: [PollOrderBy!],
  $where: PollBoolExp
) {
  poll(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    pollType {
      abbreviation
      id
      name
    }
  }
}
```

Name	Description
orderBy - [PollOrderBy!]	sort the rows by one or more columns
where - PollBoolExp	filter the rows returned

```

polls {
  ...PollFragment
}
shortName
}
rankings {
  firstPlaceVotes
  points
  poll {
    ...PollFragment
  }
  rank
  team {
    ...currentTeamsFragment
  }
}
season
seasonType
week
}
}

```

VARIABLES

```
{
  "distinctOn": ["season"],
  "limit": 987,
  "offset": 987,
  "orderBy": [PollOrderBy],
  "where": PollBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```

    "poll": [
      {
        "pollType": PollType,
        "rankings": [PollRank],
        "season": 123,
        "seasonType": season_type,
        "week": smallint
      }
    ]
  }
}

```

Queries

pollRank

fetch data from the table: "poll_rank"

Response

Returns `[PollRank!]!`

Arguments

Name	Description
<code>distinctOn - [PollRankSelectColumn]</code>	distinct select on columns

QUERY

```

query PollRank(
  $distinctOn: [PollRankSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [PollRankOrderBy!],
  $where: PollRankBoolExp
) {
  pollRank(
    distinctOn: $distinctOn,

```

Name	Description
limit - <code>Int</code>	limit the number of rows returned
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[PollRankOrderBy!]</code>	sort the rows by one or more columns
where - <code>PollRankBoolExp</code>	filter the rows returned

```

    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
) {
  firstPlaceVotes
  points
  poll {
    pollType {
      ...PollTypeFragment
    }
    rankings {
      ...PollRankFragment
    }
    season
    seasonType
    week
  }
  rank
  team {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
}
}
}

```

VARIABLES

```
{
  "distinctOn": ["firstPlaceVotes"],
  "limit": 123,
  "offset": 987,
  "orderBy": [PollRankOrderBy],
```

```
"where": PollRankBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "pollRank": [
      {
        "firstPlaceVotes": smallint,
        "points": 987,
        "poll": Poll,
        "rank": smallint,
        "team": currentTeams
      }
    ]
  }
}
```



Queries

pollType

fetch data from the table: "poll_type"

Response

Returns [PollType!]!

Arguments

Name	Description
distinctOn - [PollTypeSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PollTypeOrderBy!]	sort the rows by one or more columns
where - PollTypeBoolExp	filter the rows returned

QUERY

```
query PollType(
  $distinctOn: [PollTypeSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PollTypeOrderBy!]!,
  $where: PollTypeBoolExp
) {
  pollType(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    id
    name
    polls {
      pollType {
        ...PollTypeFragment
      }
      rankings {
        ...PollRankFragment
      }
      season
      seasonType
      week
    }
    shortName
  }
}
```

VARIABLES

```
{  
  "distinctOn": ["abbreviation"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [PollTypeOrderBy],  
  "where": PollTypeBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "pollType": [  
      {  
        "abbreviation": "abc123",  
        "id": 123,  
        "name": "xyz789",  
        "polls": [Poll],  
        "shortName": "xyz789"  
      }  
    ]  
  }  
}
```

Queries

position

fetch data from the table: "position"

Response

Returns `[Position!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[PositionSelectColumns]</code>	distinct select on columns
<code>limit</code> - <code>Int</code>	limit the number of rows returned
<code>offset</code> - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy</code> - <code>[PositionOrderBy!]</code>	sort the rows by one or more columns
<code>where</code> - <code>PositionBoolExp</code>	filter the rows returned

QUERY

```
query Position(
  $distinctOn: [PositionSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PositionOrderBy!]!,
  $where: PositionBoolExp
) {
  position(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    athletes {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
    }
  }
}
```

```
    id
    jersey
    lastName
    name
    position {
      ...PositionFragment
    }
    positionId
    recruits {
      ...RecruitFragment
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
    teamId
    weight
  }
  athletesAggregate {
    aggregate {
      ...AthleteAggregateFieldsFr
    }
    nodes {
      ...AthleteFragment
    }
  }
  displayName
  id
  name
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 123,
  "orderBy": [PositionOrderBy],
```

```
"where": PositionBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "position": [
      {
        "abbreviation": "abc123",
        "athletes": [Athlete],
        "athletesAggregate": Athlet
        "displayName": "xyz789",
        "id": smallint,
        "name": "abc123"
      }
    ]
  }
}
```



Queries

predictedPoints

fetch data from the table: "ppa"

Response

Returns [predictedPoints!]!

Arguments

Name	Description
distinctOn - [predictedPointsSel...]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [predictedPointsOrderB...]	sort the rows by one or more columns
where - predictedPointsBoolExp	filter the rows

QUERY

```
query PredictedPoints(
  $distinctOn: [predictedPointsSel...]
  $limit: Int,
  $offset: Int,
  $orderBy: [predictedPointsOrderBy!]
  $where: predictedPointsBoolExp
) {
  predictedPoints(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    distance
    down
    predictedPoints
    yardLine
  }
}
```

VARIABLES

```
{
  "distinctOn": ["distance"],
  "limit": 123,
  "offset": 123,
  "orderBy": [predictedPointsOrderBy!],
  "where": predictedPointsBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "predictedPoints": [  
      {  
        "distance": smallint,  
        "down": smallint,  
        "predictedPoints": numeric,  
        "yardLine": smallint  
      }  
    ]  
  }  
}
```



Queries

predictedPointsAggregate

fetch aggregated fields from the table:

"ppa"

Response

Returns a [predictedPointsAggregate!](#)

Arguments**QUERY**

```
query PredictedPointsAggregate(  
  $distinctOn: [predictedPointsSele  
  $limit: Int,
```

Name	Description
distinctOn - [predictedPointsSelection!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [predictedPointsOrderBy!]	sort the rows by one or more columns
where - predictedPointsBoolExp	filter the rows returned

```
$offset: Int,
$orderBy: [predictedPointsOrderBy]
$where: predictedPointsBoolExp
) {
  predictedPointsAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...predictedPointsAvgFields
      }
      count
      max {
        ...predictedPointsMaxFields
      }
      min {
        ...predictedPointsMinFields
      }
      stdDev {
        ...predictedPointsStddevFields
      }
      stdDevPop {
        ...predictedPointsStddevPop
      }
      stdDevSamp {
        ...predictedPointsStddevSam
      }
      sum {
        ...predictedPointsSumFields
      }
      varPop {
        ...predictedPointsVarPopFields
      }
      varSamp {
        ...predictedPointsVarSampFields
      }
    }
  }
}
```

```
        variance {
          ...predictedPointsVarianceF
        }
      }
    nodes {
      distance
      down
      predictedPoints
      yardLine
    }
  }
}
```

VARIABLES

```
{
  "distinctOn": ["distance"],
  "limit": 123,
  "offset": 123,
  "orderBy": [predictedPointsOrderB
  "where": predictedPointsBoolExp
}
```

RESPONSE

```
{
  "data": {
    "predictedPointsAggregate": {
      "aggregate": predictedPointsA
      "nodes": [predictedPoints]
    }
  }
}
```

```

    }
}
}
```

Queries

ratings

fetch data from the table:
"rating_systems"

Response

Returns [ratings!]!

Arguments

Name	Description
distinctOn - [ratingsSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

QUERY

```

query Ratings(
  $distinctOn: [ratingsSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [ratingsOrderBy!],
  $where: ratingsBoolExp
) {
  ratings(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    conference
    conferenceId
    elo
  }
}
```

Name	Description	
orderBy - [ratingsOrderBy!]	sort the rows by one or more columns	fpi fpiAvgWinProbabilityRank fpiDefensiveEfficiency fpiGameControlRank fpiOffensiveEfficiency fpiOverallEfficiency fpiRemainingSosRank fpiResumeRank fpiSosRank fpiSpecialTeamsEfficiency fpiStrengthOfRecordRank spDefense spOffense spOverall spSpecialTeams srs team teamId year
where - ratingsBoolExp	filter the rows returned	}

VARIABLES

```
{
  "distinctOn": ["conference"],
  "limit": 987,
  "offset": 123,
  "orderBy": [ratingsOrderBy],
  "where": ratingsBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```
"ratings": [  
  {  
    "conference": "xyz789",  
    "conferenceId": smallint,  
    "elo": 123,  
    "fpi": numeric,  
    "fpiAvgWinProbabilityRank":  
    "fpiDefensiveEfficiency": n  
    "fpiGameControlRank": small  
    "fpiOffensiveEfficiency": n  
    "fpiOverallEfficiency": num  
    "fpiRemainingSosRank": smal  
    "fpiResumeRank": smallint,  
    "fpiSosRank": smallint,  
    "fpiSpecialTeamsEfficiency"  
    "fpiStrengthOfRecordRank":  
    "spDefense": numeric,  
    "spOffense": numeric,  
    "spOverall": numeric,  
    "spSpecialTeams": numeric,  
    "srs": numeric,  
    "team": "abc123",  
    "teamId": 123,  
    "year": smallint  
  }  
]  
}
```

Queries

recruit

fetch data from the table: "recruit"

Response

Returns `[Recruit!]!`

Arguments

Name	Description
<code>distinctOn - [RecruitSelectColumn]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [RecruitOrderBy!]</code>	sort the rows by one or more columns
<code>where - RecruitBoolExp</code>	filter the rows returned

QUERY

```
query Recruit(
  $distinctOn: [RecruitSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitOrderBy!],
  $where: RecruitBoolExp
) {
  recruit(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregateFragment
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
    }
  }
}
```

```
        hometown {
            ...HometownFragment
        }
        hometownId
        id
        jersey
        lastName
        name
        position {
            ...PositionFragment
        }
        positionId
        recruits {
            ...RecruitFragment
        }
        recruitsAggregate {
            ...RecruitAggregateFragment
        }
        teamId
        weight
    }
    college {
        abbreviation
        classification
        conference
        conferenceId
        division
        school
        teamId
    }
    height
    hometown {
        athletes {
            ...AthleteFragment
        }
        athletesAggregate {
            ...AthleteAggregateFragment
        }
        city
        country
    }
```

```
countyFips
latitude
longitude
recruits {
    ...RecruitFragment
}
recruitsAggregate {
    ...RecruitAggregateFragment
}
state
}
id
name
overallRank
position {
    id
    position
    positionGroup
}
positionRank
ranking
rating
recruitSchool {
    id
    name
    recruits {
        ...RecruitFragment
    }
    recruitsAggregate {
        ...RecruitAggregateFragment
    }
}
recruitType
stars
weight
year
```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["height"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [RecruitOrderBy],  
  "where": RecruitBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "recruit": [  
      {  
        "athlete": Athlete,  
        "college": currentTeams,  
        "height": 987.65,  
        "hometown": Hometown,  
        "id": bigint,  
        "name": "xyz789",  
        "overallRank": smallint,  
        "position": RecruitPosition  
        "positionRank": smallint,  
        "ranking": smallint,  
        "rating": 987.65,  
        "recruitSchool": RecruitSch  
        "recruitType": recruit_type  
        "stars": smallint,  
        "weight": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}
```

]}
}
}

Queries

recruitAggregate

fetch aggregated fields from the table:
"recruit"

Response

Returns a [RecruitAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with

QUERY

```
query RecruitAggregate(  
  $distinctOn: [RecruitSelectColumn]  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [RecruitOrderBy!],  
  $where: RecruitBoolExp  
) {  
  recruitAggregate(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    aggregate {  
      avg {
```

Name	Description
order_by	
orderBy - [RecruitOrderBy!]	sort the rows by one or more columns
where - RecruitBoolExp	filter the rows returned

```

    ...RecruitAvgFieldsFragment
}
count
max {
    ...RecruitMaxFieldsFragment
}
min {
    ...RecruitMinFieldsFragment
}
stddev {
    ...RecruitStddevFieldsFragm
}
stddevPop {
    ...RecruitStddevPopFieldsFr
}
stddevSamp {
    ...RecruitStddevSampFieldsF
}
sum {
    ...RecruitSumFieldsFragment
}
varPop {
    ...RecruitVarPopFieldsFragm
}
varSamp {
    ...RecruitVarSampFieldsFrag
}
variance {
    ...RecruitVarianceFieldsFra
}
}
nodes {
    athlete {
        ...AthleteFragment
    }
    college {
        ...currentTeamsFragment
    }
    height
    hometown {

```

```
        ...HometownFragment
    }
    id
    name
    overallRank
    position {
        ...RecruitPositionFragment
    }
    positionRank
    ranking
    rating
    recruitSchool {
        ...RecruitSchoolFragment
    }
    recruitType
    stars
    weight
    year
}
}
```

VARIABLES

```
{
  "distinctOn": ["height"],
  "limit": 123,
  "offset": 123,
  "orderBy": [RecruitOrderBy],
  "where": RecruitBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```

    "recruitAggregate": {
      "aggregate": RecruitAggregate
      "nodes": [Recruit]
    }
  }
}

```

Queries

recruitPosition

fetch data from the table:

"recruit_position"

Response

Returns `[RecruitPosition!]!`

Arguments

Name	Description
<code>distinctOn - [RecruitPositionSelectionColumns!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned

QUERY

```

query RecruitPosition(
  $distinctOn: [RecruitPositionSele
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitPositionOrderBy
  $where: RecruitPositionBoolExp
) {
  recruitPosition(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  )
}

```

Name	Description
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitPositionOrderBy!]	sort the rows by one or more columns
where - RecruitPositionBoolExp	filter the rows

```
) {
  id
  position
  positionGroup
}
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 987,
  "offset": 123,
  "orderBy": [RecruitPositionOrderBy!],
  "where": RecruitPositionBoolExp
}
```

RESPONSE

```
{
  "data": {
    "recruitPosition": [
      {
        "id": smallint,
        "position": "abc123",
        "positionGroup": "xyz789"
      }
    ]
  }
}
```

Queries

recruitPositionAggregate

fetch aggregated fields from the table:

"recruit_position"

Response

Returns a [RecruitPositionAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitPositionSelectionColumns!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitPositionOrderBy!]	sort the rows by one or more columns

QUERY

```
query RecruitPositionAggregate(
  $distinctOn: [RecruitPositionSele
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitPositionOrderBy
  $where: RecruitPositionBoolExp
) {
  recruitPositionAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitPositionAvgFields
      }
      count
      max {
        ...RecruitPositionMaxFields
      }
      min {
        ...RecruitPositionMinFields
      }
    }
  }
}
```

Name	Description
where -	filter the rows
RecruitPositionBootcamp	returned

```
    stddev {
        ...RecruitPositionStddevFile
    }
    stddevPop {
        ...RecruitPositionStddevPop
    }
    stddevSamp {
        ...RecruitPositionStddevSam
    }
    sum {
        ...RecruitPositionSumFields
    }
    varPop {
        ...RecruitPositionVarPopFile
    }
    varSamp {
        ...RecruitPositionVarSampFile
    }
    variance {
        ...RecruitPositionVarianceFile
    }
}
nodes {
    id
    position
    positionGroup
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [RecruitPositionOrderB]
```

```
    "where": RecruitPositionBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "recruitPositionAggregate": {
      "aggregate": RecruitPositionA
      "nodes": [RecruitPosition]
    }
  }
}
```



Queries

recruitSchool

fetch data from the table:
"recruit_school"

Response

Returns [RecruitSchool!]!

QUERY

```
query RecruitSchool(
  $distinctOn: [RecruitSchoolSelect]
```

Arguments

Name	Description
distinctOn - [RecruitSchoolSelections!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitSchoolOrderBy!]	sort the rows by one or more columns
where - RecruitSchoolBoolExp	filter the rows

```
$limit: Int,
$offset: Int,
$orderBy: [RecruitSchoolOrderBy!]?
$where: RecruitSchoolBoolExp
) {
  recruitSchool(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    name
    recruits {
      athlete {
        ...AthleteFragment
      }
      college {
        ...currentTeamsFragment
      }
      height
      hometown {
        ...HometownFragment
      }
      id
      name
      overallRank
      position {
        ...RecruitPositionFragment
      }
      positionRank
      ranking
      rating
      recruitSchool {
        ...RecruitSchoolFragment
      }
      recruitType
      stars
      weight
    }
  }
}
```

```
        year
    }
  }
  recruitsAggregate {
    aggregate {
      ...RecruitAggregateFieldsFr
    }
    nodes {
      ...RecruitFragment
    }
  }
}
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 987,
  "offset": 123,
  "orderBy": [RecruitSchoolOrderBy]
  "where": RecruitSchoolBoolExp
}
```

RESPONSE

```
{
  "data": [
    "recruitSchool": [
      {
        "id": 123,
        "name": "xyz789",
        "recruits": [Recruit],
        "recruitsAggregate": Recruit
      }
    ]
  ]
}
```

```
    }  
}
```

Queries

recruitSchoolAggregate

fetch aggregated fields from the table:
"recruit_school"

Response

Returns a [RecruitSchoolAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitSchoolSelectcolumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with

QUERY

```
query RecruitSchoolAggregate(  
  $distinctOn: [RecruitSchoolSelect  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [RecruitSchoolOrderBy!]?  
  $where: RecruitSchoolBoolExp  
) {  
  recruitSchoolAggregate(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    aggregate {  
      avg {
```

Name	Description
order_by	
orderBy - [RecruitSchoolOrderBy!]	sort the rows by one or more columns
where - RecruitSchoolBoolExpr	filter the rows

```

    ...RecruitSchoolAvgFieldsFr
}
count
max {
    ...RecruitSchoolMaxFieldsFr
}
min {
    ...RecruitSchoolMinFieldsFr
}
stddev {
    ...RecruitSchoolStddevField
}
stddevPop {
    ...RecruitSchoolStddevPopFi
}
stddevSamp {
    ...RecruitSchoolStddevSampF
}
sum {
    ...RecruitSchoolSumFieldsFr
}
varPop {
    ...RecruitSchoolVarPopField
}
varSamp {
    ...RecruitSchoolVarSampFiel
}
variance {
    ...RecruitSchoolVarianceFie
}
}
nodes {
    id
    name
    recruits {
        ...RecruitFragment
    }
    recruitsAggregate {
        ...RecruitAggregateFragment
    }
}

```

```
    }  
}  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [RecruitSchoolOrderBy]  
  "where": RecruitSchoolBoolExp  
}
```



RESPONSE

```
{  
  "data": {  
    "recruitSchoolAggregate": {  
      "aggregate": RecruitSchoolAgg  
      "nodes": [RecruitSchool]  
    }  
  }  
}
```



Queries

recruitingTeam

fetch data from the table:

"recruiting_team"

Response

Returns [RecruitingTeam!]!

Arguments

Name	Description
distinctOn - [RecruitingTeamSelectColumns!]	distinct select on [RecruitingTeamSelectColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitingTeamOrderBy!]	sort the rows by one or more [RecruitingTeamOrderBy!] columns
where - RecruitingTeamBoolExp	filter the rows

QUERY

```
query RecruitingTeam(
  $distinctOn: [RecruitingTeamSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitingTeamOrderBy!]!
  $where: RecruitingTeamBoolExp
) {
  recruitingTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    points
    rank
    team {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    year
  }
}
```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [RecruitingTeamOrderBy  
  "where": RecruitingTeamBoolExp  
}]
```



RESPONSE

```
{  
  "data": {  
    "recruitingTeam": [  
      {  
        "id": 987,  
        "points": numeric,  
        "rank": smallint,  
        "team": currentTeams,  
        "year": smallint  
      }  
    ]  
  }  
}
```

recruitingTeamAggregate

fetch aggregated fields from the table:
 "recruiting_team"

Response

Returns a [RecruitingTeamAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitingTeamSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitingTeamOrderBy!]	sort the rows by one or more columns
where - RecruitingTeamBoolExp	filter the rows

QUERY

```
query RecruitingTeamAggregate(
  $distinctOn: [RecruitingTeamSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitingTeamOrderBy!]!
  $where: RecruitingTeamBoolExp
) {
  recruitingTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitingTeamAvgFieldsF
      }
      count
      max {
        ...RecruitingTeamMaxFieldsF
      }
      min {
        ...RecruitingTeamMinFieldsF
      }
      stddev {
        ...RecruitingTeamStddevFiel
      }
    }
  }
}
```

```
        stdDevPop {
          ...RecruitingTeamStddevPopF
        }
        stdDevSamp {
          ...RecruitingTeamStddevSamp
        }
        sum {
          ...RecruitingTeamSumFieldsF
        }
        varPop {
          ...RecruitingTeamVarPopFiel
        }
        varSamp {
          ...RecruitingTeamVarSampFie
        }
        variance {
          ...RecruitingTeamVarianceFi
        }
      }
      nodes {
        id
        points
        rank
        team {
          ...currentTeamsFragment
        }
        year
      }
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 987,
  "offset": 123,
  "orderBy": [RecruitingTeamOrderBy]
```

```
"where": RecruitingTeamBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "recruitingTeamAggregate": {
      "aggregate": RecruitingTeamAgg,
      "nodes": [RecruitingTeam]
    }
  }
}
```



Queries

scoreboard

fetch data from the table: "scoreboard"

Response

Returns [\[Scoreboard!\]!](#)

QUERY

```
query Scoreboard(
  $distinctOn: [ScoreboardSelectCol],
  $limit: Int,
```

Arguments

Name	Description
distinctOn - [ScoreboardSelectColumn]s	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [ScoreboardOrderBy!][]	sort the rows by one or more columns
where - ScoreboardBoolExp	filter the rows returned

```
$offset: Int,
$orderBy: [ScoreboardOrderBy!],
$where: ScoreboardBoolExp
) {
  scoreboard(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    awayClassification
    awayConference
    awayConferenceAbbreviation
    awayId
    awayLineScores
    awayPoints
    awayTeam
    city
    conferenceGame
    currentClock
    currentPeriod
    currentPossession
    currentSituation
    homeClassification
    homeConference
    homeConferenceAbbreviation
    homeId
    homeLineScores
    homePoints
    homeTeam
    id
    lastPlay
    moneylineAway
    moneylineHome
    neutralSite
    overUnder
    spread
    startDate
    startTimeTbd
  }
}
```

```
        state
        status
        temperature
        tv
        venue
        weatherDescription
        windDirection
        windSpeed
    }
}
```

VARIABLES

```
{
  "distinctOn": ["awayClassification"],
  "limit": 123,
  "offset": 987,
  "orderBy": [ScoreboardOrderBy],
  "where": ScoreboardBoolExp
}
```

RESPONSE

```
{
  "data": {
    "scoreboard": [
      {
        "awayClassification": divis,
        "awayConference": "xyz789",
        "awayConferenceAbbreviation": "xyz789",
        "awayId": 987,
        "awayLineScores": [smallint],
        "awayPoints": smallint,
        "awayTeam": "abc123",
        "city": "xyz789",
        "division": "xyz789",
        "homeClassification": "xyz789",
        "homeConference": "xyz789",
        "homeConferenceAbbreviation": "xyz789",
        "homeId": 987,
        "homeLineScores": [smallint],
        "homePoints": smallint,
        "homeTeam": "abc123",
        "id": 987,
        "isNeutral": false,
        "neutralSite": null,
        "neutralSiteCity": null,
        "neutralSiteState": null,
        "neutralSiteZip": null,
        "neutralSiteName": null,
        "neutralSiteAddress": null,
        "neutralSiteLatitude": null,
        "neutralSiteLongitude": null,
        "neutralSiteAddress2": null,
        "neutralSiteCity2": null,
        "neutralSiteState2": null,
        "neutralSiteZip2": null,
        "neutralSiteName2": null,
        "neutralSiteAddress2": null,
        "neutralSiteLatitude2": null,
        "neutralSiteLongitude2": null,
        "neutralSiteAddress3": null,
        "neutralSiteCity3": null,
        "neutralSiteState3": null,
        "neutralSiteZip3": null,
        "neutralSiteName3": null,
        "neutralSiteAddress3": null,
        "neutralSiteLatitude3": null,
        "neutralSiteLongitude3": null
      }
    ]
  }
}
```

```
        "conferenceGame": false,
        "currentClock": "abc123",
        "currentPeriod": smallint,
        "currentPossession": "xyz78"
      "currentSituation": "xyz789"
      "homeClassification": divis
      "homeConference": "xyz789",
      "homeConferenceAbbreviation"
      "homeId": 123,
      "homeLineScores": [smallint
      "homePoints": smallint,
      "homeTeam": "abc123",
      "id": 123,
      "lastPlay": "xyz789",
      "moneylineAway": 987,
      "moneylineHome": 987,
      "neutralSite": true,
      "overUnder": numeric,
      "spread": numeric,
      "startDate": timestamptz,
      "startTimeTbd": false,
      "state": "xyz789",
      "status": game_status,
      "temperature": numeric,
      "tv": "xyz789",
      "venue": "abc123",
      "weatherDescription": "xyz789"
      "windDirection": numeric,
      "windSpeed": numeric
    }
  ]
}
```

Queries

teamTalent

fetch data from the table: "team_talent"

Response

Returns `[TeamTalent!]!`

Arguments

Name	Description
<code>distinctOn - [TeamTalentSelectColumn!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [TeamTalentOrderBy!]</code>	sort the rows by one or more columns
<code>where - TeamTalentBoolExp</code>	filter the rows returned

QUERY

```
query TeamTalent(
  $distinctOn: [TeamTalentSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [TeamTalentOrderBy!],
  $where: TeamTalentBoolExp
) {
  teamTalent(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    talent
    team {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    year
  }
}
```

{
}**VARIABLES**{
 "distinctOn": ["talent"],
 "limit": 987,
 "offset": 123,
 "orderBy": [TeamTalentOrderBy],
 "where": TeamTalentBoolExp
}**RESPONSE**{
 "data": {
 "teamTalent": [
 {
 "talent": numeric,
 "team": currentTeams,
 "year": smallint
 }
]
 }
}

Queries

teamTalentAggregate

fetch aggregated fields from the table:

"team_talent"

Response

Returns a [TeamTalentAggregate!](#)

Arguments

Name	Description
distinctOn - [TeamTalentSelectColumn!]s	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [TeamTalentOrderBy!]!	sort the rows by one or more columns
where - TeamTalentBoolExp	filter the rows returned

QUERY

```
query TeamTalentAggregate(
  $distinctOn: [TeamTalentSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [TeamTalentOrderBy!],
  $where: TeamTalentBoolExp
) {
  teamTalentAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...TeamTalentAvgFieldsFragm
      }
      count
      max {
        ...TeamTalentMaxFieldsFragm
      }
      min {
        ...TeamTalentMinFieldsFragm
      }
      stddev {
        ...TeamTalentStddevFieldsFr
      }
    }
  }
}
```

```

    stdDevPop {
      ...TeamTalentStdDevPopField
    }
    stdDevSamp {
      ...TeamTalentStdDevSampField
    }
    sum {
      ...TeamTalentSumFieldsFragment
    }
    varPop {
      ...TeamTalentVarPopFieldsFragment
    }
    varSamp {
      ...TeamTalentVarSampFieldsFragment
    }
    variance {
      ...TeamTalentVarianceFields
    }
  }
  nodes {
    talent
    team {
      ...currentTeamsFragment
    }
    year
  }
}

```

VARIABLES

```
{
  "distinctOn": ["talent"],
  "limit": 987,
  "offset": 987,
  "orderBy": [TeamTalentOrderBy],
}
```

```
    "where": TeamTalentBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "teamTalentAggregate": {
      "aggregate": TeamTalentAggregat
      "nodes": [TeamTalent]
    }
  }
}
```



Queries

transfer

fetch data from the table: "transfer"

Response

Returns `[Transfer!]!`

Arguments

QUERY

```
query Transfer(
  $distinctOn: [TransferSelectColumn]
  $limit: Int,
  $offset: Int,
```

Name	Description
distinctOn - [TransferSelectColumn]	distinct select on [TransferSelectColumn]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [TransferOrderBy!]	sort the rows by one or more columns
where - TransferBoolExp	filter the rows returned

```
$orderBy: [TransferOrderBy!],
$where: TransferBoolExp
) {
  transfer(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    eligibility
    firstName
    fromTeam {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    lastName
    position {
      id
      position
      positionGroup
    }
    rating
    season
    stars
    toTeam {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    transferDate
  }
}
```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["eligibility"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [TransferOrderBy],  
  "where": TransferBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "transfer": [  
      {  
        "eligibility": "abc123",  
        "firstName": "abc123",  
        "fromTeam": currentTeams,  
        "lastName": "abc123",  
        "position": RecruitPosition,  
        "rating": numeric,  
        "season": smallint,  
        "stars": smallint,  
        "toTeam": currentTeams,  
        "transferDate": timestamp  
      }  
    ]  
  }  
}
```

Queries

weatherCondition

fetch data from the table:

"weather_condition"

Response

Returns `[WeatherCondition!]!`

Arguments

Name	Description
<code>distinctOn - [WeatherConditionSelectionColumn!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [WeatherConditionOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
query WeatherCondition(
  $distinctOn: [WeatherConditionSel
  $limit: Int,
  $offset: Int,
  $orderBy: [WeatherConditionOrderB
  $where: WeatherConditionBoolExp
) {
  weatherCondition(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    description
    id
  }
}
```

VARIABLES

Name	Description
where -	filter the rows
WeatherConditionBoolExp	

```
{  
  "distinctOn": ["description"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [WeatherConditionOrder  
  "where": WeatherConditionBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "weatherCondition": [  
      {  
        "description": "abc123",  
        "id": smallint  
      }  
    ]  
  }  
}
```

Subscriptions

adjustedPlayerMetrics

An array relationship

Response

Returns [AdjustedPlayerMetrics!]!

Arguments

Name	Description
distinctOn - [AdjustedPlayerMetrics!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AdjustedPlayerMetricsOrderBy!]	sort the rows by one or more columns
where - AdjustedPlayerMetricsFilterExp	filter the rows

QUERY

```

subscription AdjustedPlayerMetrics(
  $distinctOn: [AdjustedPlayerMetrics!]
  $limit: Int,
  $offset: Int,
  $orderBy: [AdjustedPlayerMetricsOrderBy!]
  $where: AdjustedPlayerMetricsBool
) {
  adjustedPlayerMetrics(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
    }
    adjustedPlayerMetricsAggregate {
      ...AdjustedPlayerMetricsAggregateFragment
    }
    athleteTeams {
      ...AthleteTeamFragment
    }
    athleteTeamsAggregate {
      ...AthleteTeamAggregateFragment
    }
    firstName
    height
    hometown {
      ...HometownFragment
    }
    hometownId
    id
    jersey
    lastName
    name
  }
}

```

```
        position {
          ...PositionFragment
        }
        positionId
        recruits {
          ...RecruitFragment
        }
        recruitsAggregate {
          ...RecruitAggregateFragment
        }
        teamId
        weight
      }
      athleteId
      metricType
      metricValue
      plays
      year
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [AdjustedPlayerMetrics
  "where": AdjustedPlayerMetricsBoo
}
```

RESPONSE

```
{
  "data": {
```

```
"adjustedPlayerMetrics": [  
  {  
    "athlete": Athlete,  
    "athleteId": bigint,  
    "metricType": player_adjust  
    "metricValue": numeric,  
    "plays": smallint,  
    "year": smallint  
  }  
]  
}  
}
```

Subscriptions

adjustedPlayerMetricsAggregate

An aggregate relationship

Response

Returns an
[AdjustedPlayerMetricsAggregate!](#)

Arguments

QUERY

```
subscription AdjustedPlayerMetricsA  
  $distinctOn: [AdjustedPlayerMetricsAggregate!]!  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [AdjustedPlayerMetricsOrder!]!  
  $where: AdjustedPlayerMetricsBool  
) {
```

Name	Description
distinctOn - [AdjustedPlayerMetricsColumn!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AdjustedPlayerMetricsOrderBy!]	sort the rows by one or more columns
where - AdjustedPlayerMetricsWhereExp	filter the rows

```

adjustedPlayerMetricsAggregate(
  distinctOn: $distinctOn,
  limit: $limit,
  offset: $offset,
  orderBy: $orderBy,
  where: $where
) {
  aggregate {
    avg {
      ...AdjustedPlayerMetricsAvg
    }
    count
    max {
      ...AdjustedPlayerMetricsMax
    }
    min {
      ...AdjustedPlayerMetricsMin
    }
    stddev {
      ...AdjustedPlayerMetricsStd
    }
    stddevPop {
      ...AdjustedPlayerMetricsStd
    }
    stddevSamp {
      ...AdjustedPlayerMetricsStd
    }
    sum {
      ...AdjustedPlayerMetricsSum
    }
    varPop {
      ...AdjustedPlayerMetricsVar
    }
    varSamp {
      ...AdjustedPlayerMetricsVar
    }
    variance {
      ...AdjustedPlayerMetricsVar
    }
  }
}

```

```
  nodes {
    athlete {
      ...AthleteFragment
    }
    athleteId
    metricType
    metricValue
    plays
    year
  }
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [AdjustedPlayerMetrics
  "where": AdjustedPlayerMetricsBoo
}
```

RESPONSE

```
{
  "data": {
    "adjustedPlayerMetricsAggregate": [
      "aggregate": AdjustedPlayerMe
      "nodes": [AdjustedPlayerMetri
    ]
  }
}
```

```
}
```

Subscriptions

adjustedTeamMetrics

fetch data from the table:
"adjusted_team_metrics"

Response

Returns [\[AdjustedTeamMetrics!\]!](#)

Arguments

Name	Description
distinctOn - [AdjustedTeamMetrics!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

QUERY

```
subscription AdjustedTeamMetrics(
  $distinctOn: [AdjustedTeamMetrics
  $limit: Int,
  $offset: Int,
  $orderBy: [AdjustedTeamMetricsOrd
  $where: AdjustedTeamMetricsBoolEx
) {
  adjustedTeamMetrics(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    epa
    epaAllowed
    explosiveness
  }
}
```

Name	Description	
orderBy - [AdjustedTeamMetricsorderBy!]	sort the rows by one or more columns	explosivenessAllowed highlightYards highlightYardsAllowed lineYards lineYardsAllowed openFieldYards openFieldYardsAllowed passingDownsSuccess passingDownsSuccessAllowed passingEpa passingEpaAllowed rushingEpa rushingEpaAllowed secondLevelYards secondLevelYardsAllowed standardDownsSuccess standardDownsSuccessAllowed success successAllowed team { abbreviation classification conference conferenceId division school teamId } teamId year }
where - AdjustedTeamMetricsWhereInput	filter the rows	

VARIABLES

```
{
  "distinctOn": ["epa"],
  "limit": 123,
```

```
        "offset": 123,  
        "orderBy": [AdjustedTeamMetricsOr  
        "where": AdjustedTeamMetricsBoole  
    }]
```

RESPONSE

```
{  
  "data": {  
    "adjustedTeamMetrics": [  
      {  
        "epa": numeric,  
        "epaAllowed": numeric,  
        "explosiveness": numeric,  
        "explosivenessAllowed": num  
        "highlightYards": numeric,  
        "highlightYardsAllowed": nu  
        "lineYards": numeric,  
        "lineYardsAllowed": numeric  
        "openFieldYards": numeric,  
        "openFieldYardsAllowed": nu  
        "passingDownsSuccess": num  
        "passingDownsSuccessAllowed  
        "passingEpa": numeric,  
        "passingEpaAllowed": numeri  
        "rushingEpa": numeric,  
        "rushingEpaAllowed": numeri  
        "secondLevelYards": numeric  
        "secondLevelYardsAllowed":  
        "standardDownsSuccess": num  
        "standardDownsSuccessAllowe  
        "success": numeric,  
        "successAllowed": numeric,  
        "team": currentTeams,  
        "teamId": 123,  
        "year": smallint  
      }  
    ]  
  }  
}
```

{
}

Subscriptions

adjustedTeamMetricsAggregate

fetch aggregated fields from the table:

"adjusted_team_metrics"

Response

Returns an
AdjustedTeamMetricsAggregate!

Arguments

Name	Description
<code>distinctOn - [AdjustedTeamMetricsColumn!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned

QUERY

```
subscription AdjustedTeamMetricsAgg
  $distinctOn: [AdjustedTeamMetrics
  $limit: Int,
  $offset: Int,
  $orderBy: [AdjustedTeamMetricsOrd
  $where: AdjustedTeamMetricsBoolEx
) {
  adjustedTeamMetricsAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[AdjustedTeamMetricsOrderBy!]</code>	sort the rows by one or more columns
where - <code>AdjustedTeamMetricsWhereInput</code>	filter the rows

```

avg {
  ...AdjustedTeamMetricsAvgFi
}
count
max {
  ...AdjustedTeamMetricsMaxFi
}
min {
  ...AdjustedTeamMetricsMinFi
}
stddev {
  ...AdjustedTeamMetricsStdde
}
stddevPop {
  ...AdjustedTeamMetricsStdde
}
stddevSamp {
  ...AdjustedTeamMetricsStdde
}
sum {
  ...AdjustedTeamMetricsSumFi
}
varPop {
  ...AdjustedTeamMetricsVarPo
}
varSamp {
  ...AdjustedTeamMetricsVarSa
}
variance {
  ...AdjustedTeamMetricsVaria
}
}
nodes {
  epa
  epaAllowed
  explosiveness
  explosivenessAllowed
  highlightYards
  highlightYardsAllowed
  lineYards
}

```

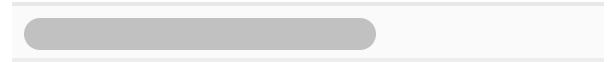
```
        lineYardsAllowed
        openFieldYards
        openFieldYardsAllowed
        passingDownsSuccess
        passingDownsSuccessAllowed
        passingEpa
        passingEpaAllowed
        rushingEpa
        rushingEpaAllowed
        secondLevelYards
        secondLevelYardsAllowed
        standardDownsSuccess
        standardDownsSuccessAllowed
        success
        successAllowed
      team {
        ...currentTeamsFragment
      }
      teamId
      year
    }
  }
}
```

VARIABLES

```
{
  "distinctOn": ["epa"],
  "limit": 123,
  "offset": 123,
  "orderBy": [AdjustedTeamMetricsOr
  "where": AdjustedTeamMetricsBoole
}
```

RESPONSE

```
{  
  "data": {  
    "adjustedTeamMetricsAggregate": {  
      "aggregate": AdjustedTeamMet  
      "nodes": [AdjustedTeamMetrics  
    }  
  }  
}
```



Subscriptions

athlete

fetch data from the table: "athlete"

Response

Returns [\[Athlete!\]!](#)

Arguments

Name	Description
<code>distinctOn - [AthleteSelectColumn]!columns</code>	distinct select on [AthleteSelectColumn]!columns

QUERY

```
subscription Athlete(  
  $distinctOn: [AthleteSelectColumn  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [AthleteOrderBy!],  
  $where: AthleteBoolExp  
) {  
  athlete(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
)  
}
```

Name	Description
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [AthleteOrderBy!]	sort the rows by one or more columns
where - AthleteBoolExp	filter the rows returned

```

limit: $limit,
offset: $offset,
orderBy: $orderBy,
where: $where
) {
  adjustedPlayerMetrics {
    athlete {
      ...AthleteFragment
    }
    athleteId
    metricType
    metricValue
    plays
    year
  }
  adjustedPlayerMetricsAggregate {
    aggregate {
      ...AdjustedPlayerMetricsAgg
    }
    nodes {
      ...AdjustedPlayerMetricsFra
    }
  }
  athleteTeams {
    athlete {
      ...AthleteFragment
    }
    athleteId
    endYear
    startYear
    team {
      ...historicalTeamFragment
    }
    teamId
  }
  athleteTeamsAggregate {
    aggregate {
      ...AthleteTeamAggregateFiel
    }
    nodes {
  
```

```
    ...AthleteTeamFragment
  }
}
firstName
height
hometown {
  athletes {
    ...AthleteFragment
  }
  athletesAggregate {
    ...AthleteAggregateFragment
  }
  city
  country
  countyFips
  latitude
  longitude
  recruits {
    ...RecruitFragment
  }
  recruitsAggregate {
    ...RecruitAggregateFragment
  }
  state
}
hometownId
id
jersey
lastName
name
position {
  abbreviation
  athletes {
    ...AthleteFragment
  }
  athletesAggregate {
    ...AthleteAggregateFragment
  }
  displayName
  id
```

```
        name
    }
    positionId
    recruits {
        athlete {
            ...AthleteFragment
        }
        college {
            ...currentTeamsFragment
        }
        height
        hometown {
            ...HometownFragment
        }
        id
        name
        overallRank
        position {
            ...RecruitPositionFragment
        }
        positionRank
        ranking
        rating
        recruitSchool {
            ...RecruitSchoolFragment
        }
        recruitType
        stars
        weight
        year
    }
    recruitsAggregate {
        aggregate {
            ...RecruitAggregateFieldsFr
        }
        nodes {
            ...RecruitFragment
        }
    }
    teamId
```

```
        weight
    }
}
```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 987,
  "offset": 987,
  "orderBy": [AthleteOrderBy],
  "where": AthleteBoolExp
}
```

RESPONSE

```
{
  "data": {
    "athlete": [
      {
        "adjustedPlayerMetrics": [A
          "adjustedPlayerMetricsAggre
          "athleteTeams": [AthleteTea
          "athleteTeamsAggregate": At
          "firstName": "abc123",
          "height": smallint,
          "hometown": Hometown,
          "hometownId": 123,
          "id": bigint,
          "jersey": smallint,
          "lastName": "abc123",
          "name": "xyz789",
          "position": Position,
          "positionId": smallint,
          "recruits": [Recruit],
          "recruitsAggregate": Recrui
        ]
      }
    ]
  }
}
```

```

        "teamId": 987,
        "weight": smallint
    }
]
}
}
}
```

Subscriptions

athleteAggregate

fetch aggregated fields from the table:
"athlete"

Response

Returns an [AthleteAggregate!](#)

Arguments

Name	Description
<code>distinctOn - [AthleteSelectColumn]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned

QUERY

```

subscription AthleteAggregate(
  $distinctOn: [AthleteSelectColumn],
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteOrderBy!],
  $where: AthleteBoolExp
) {
  athleteAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  )
}
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[AthleteOrderBy!]</code>	sort the rows by one or more columns
where - <code>AthleteBoolExp</code>	filter the rows returned

```
) {
  aggregate {
    avg {
      ...AthleteAvgFieldsFragment
    }
    count
    max {
      ...AthleteMaxFieldsFragment
    }
    min {
      ...AthleteMinFieldsFragment
    }
    stddev {
      ...AthleteStddevFieldsFragm
    }
    stddevPop {
      ...AthleteStddevPopFieldsFr
    }
    stddevSamp {
      ...AthleteStddevSampFieldsF
    }
    sum {
      ...AthleteSumFieldsFragment
    }
    varPop {
      ...AthleteVarPopFieldsFragm
    }
    varSamp {
      ...AthleteVarSampFieldsFrag
    }
    variance {
      ...AthleteVarianceFieldsFra
    }
  }
  nodes {
    adjustedPlayerMetrics {
      ...AdjustedPlayerMetricsFra
    }
    adjustedPlayerMetricsAggregat
    ...AdjustedPlayerMetricsAgg
  }
}
```

```
        }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFrag
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
      positionId
      recruits {
        ...RecruitFragment
      }
      recruitsAggregate {
        ...RecruitAggregateFragment
      }
      teamId
      weight
    }
  }
}
```

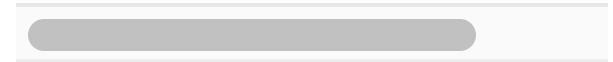
VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 987,
```

```
"offset": 123,  
"orderBy": [AthleteOrderBy],  
"where": AthleteBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "athleteAggregate": {  
      "aggregate": AthleteAggregate  
      "nodes": [Athlete]  
    }  
  }  
}
```



Subscriptions

athleteByPk

fetch data from the table: "athlete" using primary key columns

Response

QUERY

Returns an [Athlete](#)

Arguments

Name	Description
id - <code>bigint!</code>	

```

subscription AthleteByPk($id: bigin
  athleteByPk(id: $id) {
    adjustedPlayerMetrics {
      athlete {
        ...AthleteFragment
      }
      athleteId
      metricType
      metricValue
      plays
      year
    }
    adjustedPlayerMetricsAggregate {
      aggregate {
        ...AdjustedPlayerMetricsAgg
      }
      nodes {
        ...AdjustedPlayerMetricsFra
      }
    }
    athleteTeams {
      athlete {
        ...AthleteFragment
      }
      athleteId
      endYear
      startYear
      team {
        ...historicalTeamFragment
      }
      teamId
    }
    athleteTeamsAggregate {
      aggregate {
        ...AthleteTeamAggregateFiel
      }
      nodes {
        ...AthleteTeamFragment
      }
    }
  }
}

```

```
        }
      firstName
      height
      hometown {
        athletes {
          ...AthleteFragment
        }
        athletesAggregate {
          ...AthleteAggregateFragment
        }
        city
        country
        countyFips
        latitude
        longitude
        recruits {
          ...RecruitFragment
        }
        recruitsAggregate {
          ...RecruitAggregateFragment
        }
        state
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        abbreviation
        athletes {
          ...AthleteFragment
        }
        athletesAggregate {
          ...AthleteAggregateFragment
        }
        displayName
        id
        name
      }
    }
```

```
positionId
recruits {
  athlete {
    ...AthleteFragment
  }
  college {
    ...currentTeamsFragment
  }
  height
  hometown {
    ...HometownFragment
  }
  id
  name
  overallRank
  position {
    ...RecruitPositionFragment
  }
  positionRank
  ranking
  rating
  recruitSchool {
    ...RecruitSchoolFragment
  }
  recruitType
  stars
  weight
  year
}
recruitsAggregate {
  aggregate {
    ...RecruitAggregateFieldsFr
  }
  nodes {
    ...RecruitFragment
  }
}
teamId
weight
```

```
    }  
}
```

VARIABLES

```
{"id": bigint}
```

RESPONSE

```
{  
  "data": {  
    "athleteByPk": {  
      "adjustedPlayerMetrics": [Adj  
      "adjustedPlayerMetricsAggrega  
      "athleteTeams": [AthleteTeam]  
      "athleteTeamsAggregate": Athl  
      "firstName": "xyz789",  
      "height": smallint,  
      "hometown": Hometown,  
      "hometownId": 123,  
      "id": bigint,  
      "jersey": smallint,  
      "lastName": "abc123",  
      "name": "abc123",  
      "position": Position,  
      "positionId": smallint,  
      "recruits": [Recruit],  
      "recruitsAggregate": RecruitA  
      "teamId": 987,  
      "weight": smallint  
    }  
  }  
}
```

Subscriptions

athleteTeam

fetch data from the table:

"athlete_team"

Response

Returns `[AthleteTeam!]!`

Arguments

Name	Description
<code>distinctOn - [AthleteTeamSelectColumns]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [AthleteTeamOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
subscription AthleteTeam(
  $distinctOn: [AthleteTeamSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteTeamOrderBy!],
  $where: AthleteTeamBoolExp
) {
  athleteTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFra
      }
      adjustedPlayerMetricsAggregat
        ...AdjustedPlayerMetricsAgg
    }
    athleteTeams {
      ...AthleteTeamFragment
    }
    athleteTeamsAggregate {

```

Name	Description	
where -	filter the rows	<pre> ...AthleteTeamAggregateFrag } firstName height hometown { ...HometownFragment } hometownId id jersey lastName name position { ...PositionFragment } positionId recruits { ...RecruitFragment } recruitsAggregate { ...RecruitAggregateFragment } teamId weight } athleteId endYear startYear team { abbreviation active altColor altName classification color conference conferenceAbbreviation conferenceId conferenceShortName countryCode </pre>

```
        displayName
        division
        endYear
        id
        images
        mascot
        ncaaName
        nickname
        school
        shortDisplayName
        startYear
        twitter
    }
    teamId
}
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 123,
  "offset": 123,
  "orderBy": [AthleteTeamOrderBy],
  "where": AthleteTeamBoolExp
}
```

RESPONSE

```
{
  "data": {
    "athleteTeam": [
      {
        "athlete": Athlete,
        "athleteId": bigint,
      }
    ]
  }
}
```

```

    "endYear": smallint,
    "startYear": smallint,
    "team": historicalTeam,
    "teamId": 123
  }
]
}
}
}

```

Subscriptions

athleteTeamAggregate

fetch aggregated fields from the table:

"athlete_team"

Response

Returns an [AthleteTeamAggregate!](#)

Arguments

Name	Description
distinctOn - [AthleteTeamSelectColumns]	distinct select on

QUERY

```

subscription AthleteTeamAggregate(
  $distinctOn: [AthleteTeamSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [AthleteTeamOrderBy!],
  $where: AthleteTeamBoolExp
) {
  athleteTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
  )
}

```

Name	Description
limit - <code>Int</code>	limit the number of rows returned
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[AthleteTeamOrderBy!]</code>	sort the rows by one or more columns
where - <code>AthleteTeamBoolExp</code>	filter the rows returned

```

        orderBy: $orderBy,
        where: $where
    ) {
        aggregate {
            avg {
                ...AthleteTeamAvgFieldsFrag
            }
            count
            max {
                ...AthleteTeamMaxFieldsFrag
            }
            min {
                ...AthleteTeamMinFieldsFrag
            }
            stdDev {
                ...AthleteTeamStddevFieldsFrag
            }
            stdDevPop {
                ...AthleteTeamStddevPopFieldsFrag
            }
            stdDevSamp {
                ...AthleteTeamStddevSampFieldsFrag
            }
            sum {
                ...AthleteTeamSumFieldsFrag
            }
            varPop {
                ...AthleteTeamVarPopFieldsFrag
            }
            varSamp {
                ...AthleteTeamVarSampFieldsFrag
            }
            variance {
                ...AthleteTeamVarianceFieldsFrag
            }
        }
        nodes {
            athlete {
                ...AthleteFragment
            }
        }
    }
}

```

```
    athleteId  
    endYear  
    startYear  
    team {  
      ...historicalTeamFragment  
    }  
    teamId  
  }  
}
```

VARIABLES

```
{  
  "distinctOn": ["athleteId"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [AthleteTeamOrderBy],  
  "where": AthleteTeamBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "athleteTeamAggregate": {  
      "aggregate": AthleteTeamAggre  
      "nodes": [AthleteTeam]  
    }  
  }  
}
```

Subscriptions

calendar

fetch data from the table: "calendar"

Response

Returns `[Calendar!]!`

Arguments

Name	Description
<code>distinctOn - [CalendarSelectColumns]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [CalendarOrderBy!]</code>	sort the rows by one or more columns
<code>where - CalendarBoolExp</code>	filter the rows returned

QUERY

```
subscription Calendar(
  $distinctOn: [CalendarSelectColum
  $limit: Int,
  $offset: Int,
  $orderBy: [CalendarOrderBy!],
  $where: CalendarBoolExp
) {
  calendar(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    endDate
    seasonType
    startDate
    week
    year
  }
}
```

VARIABLES

```
{  
  "distinctOn": ["endDate"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [CalendarOrderBy],  
  "where": CalendarBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "calendar": [  
      {  
        "endDate": timestamp,  
        "seasonType": season_type,  
        "startDate": timestamp,  
        "week": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}
```



Subscriptions

coach

fetch data from the table: "coach"

Response

Returns `[Coach!]!`

Arguments

Name	Description
<code>distinctOn - [CoachSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [CoachOrderBy!]</code>	sort the rows by one or more columns
<code>where - CoachBoolExp</code>	filter the rows returned

QUERY

```
subscription Coach(
  $distinctOn: [CoachSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachOrderBy!],
  $where: CoachBoolExp
) {
  coach(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    firstName
    id
    lastName
    seasons {
      coach {
        ...CoachFragment
      }
      games
      losses
      postseasonRank
      preseasonRank
      team {
        ...currentTeamsFragment
      }
      ties
      wins
      year
    }
    seasonsAggregate {
      aggregate {

```

```
        ...CoachSeasonAggregateField
    }
  nodes {
    ...CoachSeasonFragment
  }
}
}
```

VARIABLES

```
{
  "distinctOn": ["firstName"],
  "limit": 123,
  "offset": 987,
  "orderBy": [CoachOrderBy],
  "where": CoachBoolExp
}
```

RESPONSE

```
{
  "data": {
    "coach": [
      {
        "firstName": "abc123",
        "id": 123,
        "lastName": "abc123",
        "seasons": [CoachSeason],
        "seasonsAggregate": CoachSe
      }
    ]
  }
}
```

Subscriptions

coachAggregate

fetch aggregated fields from the table:
 "coach"

Response

Returns a [CoachAggregate!](#)

Arguments

Name	Description
distinctOn - [CoachSelectColumn!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [CoachOrderBy!]	sort the rows by one or more

QUERY

```
subscription CoachAggregate(
  $distinctOn: [CoachSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachOrderBy!]!,
  $where: CoachBoolExp
) {
  coachAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...CoachAvgFieldsFragment
      }
      count
      max {
```

Name	Description
	columns
where - CoachBoolExp	filter the rows returned

```

    ...CoachMaxFieldsFragment
}
min {
  ...CoachMinFieldsFragment
}
stddev {
  ...CoachStddevFieldsFragmen
}
stddevPop {
  ...CoachStddevPopFieldsFrag
}
stddevSamp {
  ...CoachStddevSampFieldsFra
}
sum {
  ...CoachSumFieldsFragment
}
varPop {
  ...CoachVarPopFieldsFragmen
}
varSamp {
  ...CoachVarSampFieldsFragme
}
variance {
  ...CoachVarianceFieldsFragm
}
}
nodes {
  firstName
  id
  lastName
  seasons {
    ...CoachSeasonFragment
  }
  seasonsAggregate {
    ...CoachSeasonAggregateFrag
  }
}

```

{
}**VARIABLES**{
 "distinctOn": ["firstName"],
 "limit": 123,
 "offset": 123,
 "orderBy": [CoachOrderBy],
 "where": CoachBoolExp
}
}**RESPONSE**{
 "data": {
 "coachAggregate": {
 "aggregate": CoachAggregateFi
 "nodes": [Coach]
 }
 }
}


Subscriptions

coachSeason

fetch data from the table:
"coach_season"

Response

Returns `[CoachSeason!]!`

Arguments

Name	Description
<code>distinctOn - [CoachSeasonSelectColumns]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [CoachSeasonOrderBy!]</code>	sort the rows by one or more columns
<code>where - CoachSeasonBoolExp</code>	filter the rows returned

QUERY

```
subscription CoachSeason(
  $distinctOn: [CoachSeasonSelectCo
  $limit: Int,
  $offset: Int,
  $orderBy: [CoachSeasonOrderBy!],
  $where: CoachSeasonBoolExp
) {
  coachSeason(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    coach {
      firstName
      id
      lastName
      seasons {
        ...CoachSeasonFragment
      }
      seasonsAggregate {
        ...CoachSeasonAggregateFrag
      }
    }
    games
    losses
    postseasonRank
    preseasonRank
    team {
      abbreviation
      classification
      conference
    }
  }
}
```

```
        conferenceId
        division
        school
        teamId
    }
    ties
    wins
    year
}
}
```

VARIABLES

```
{
  "distinctOn": ["games"],
  "limit": 987,
  "offset": 123,
  "orderBy": [CoachSeasonOrderBy],
  "where": CoachSeasonBoolExp
}
```

RESPONSE

```
{
  "data": [
    {
      "coach": Coach,
      "games": smallint,
      "losses": smallint,
      "postseasonRank": smallint,
      "preseasonRank": smallint,
      "team": currentTeams,
      "ties": smallint,
      "wins": smallint,
    }
  ]
}
```

```
        "year": smallint  
    }  
}  
]  
}  
}
```

Subscriptions

coachSeasonAggregate

fetch aggregated fields from the table:

"coach season"

Response

Returns a *CoachSeasonAggregate*!

Arguments

Name	Description
distinctOn - [CoachSeasonSelectColumns]	distinct select on
limit - Int	limit the number of rows returned

QUERY

```
subscription CoachSeasonAggregate(  
    $distinctOn: [CoachSeasonSelectCo  
    $limit: Int,  
    $offset: Int,  
    $orderBy: [CoachSeasonOrderBy!],  
    $where: CoachSeasonBoolExp  
) {  
    coachSeasonAggregate(  
        distinctOn: $distinctOn,  
        limit: $limit,  
        offset: $offset,  
        orderBy: $orderBy,
```

Name	Description
offset - Int	skip the first n rows. Use only with order_by
orderBy - [CoachSeasonOrderBy!]!	sort the rows by one or more columns
where - CoachSeasonBoolExp	filter the rows returned

```

  where: $where
} {
  aggregate {
    avg {
      ...CoachSeasonAvgFieldsFrag
    }
    count
    max {
      ...CoachSeasonMaxFieldsFrag
    }
    min {
      ...CoachSeasonMinFieldsFrag
    }
    stddev {
      ...CoachSeasonStddevFieldsF
    }
    stddevPop {
      ...CoachSeasonStddevPopFiel
    }
    stddevSamp {
      ...CoachSeasonStddevSampFie
    }
    sum {
      ...CoachSeasonSumFieldsFrag
    }
    varPop {
      ...CoachSeasonVarPopFieldsF
    }
    varSamp {
      ...CoachSeasonVarSampFields
    }
    variance {
      ...CoachSeasonVarianceField
    }
  }
  nodes {
    coach {
      ...CoachFragment
    }
    games
  }
}

```

```
  losses
  postseasonRank
  preseasonRank
  team {
    ...currentTeamsFragment
  }
  ties
  wins
  year
}
}
```

VARIABLES

```
{
  "distinctOn": ["games"],
  "limit": 123,
  "offset": 987,
  "orderBy": [CoachSeasonOrderBy],
  "where": CoachSeasonBoolExp
}
```

RESPONSE

```
{
  "data": {
    "coachSeasonAggregate": {
      "aggregate": CoachSeasonAggre
      "nodes": [CoachSeason]
    }
  }
}
```

```

    }
}
}
```

Subscriptions

conference

fetch data from the table: "conference"

Response

Returns [\[Conference!\]!](#)

Arguments

Name	Description
distinctOn - [ConferenceSelectColumn]s	distinct select on [ConferenceSelectColumn]s
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with <code>order_by</code>

QUERY

```

subscription Conference(
  $distinctOn: [ConferenceSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [ConferenceOrderBy!],
  $where: ConferenceBoolExp
) {
  conference(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    division
    id
    name
  }
}
```

Name	Description
orderBy - [ConferenceOrderBy!]	sort the rows by one or more columns
where - ConferenceBoolExp	filter the rows returned

shortName
srName

}

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 987,
  "orderBy": [ConferenceOrderBy],
  "where": ConferenceBoolExp
}
```

RESPONSE

```
{
  "data": {
    "conference": [
      {
        "abbreviation": "xyz789",
        "division": division,
        "id": smallint,
        "name": "xyz789",
        "shortName": "xyz789",
        "srName": "xyz789"
      }
    ]
  }
}
```

Subscriptions

currentTeams

fetch data from the table:

"current_conferences"

Response

Returns `[currentTeams!]!`

Arguments

Name	Description
<code>distinctOn - [currentTeamsSelectColumns!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [currentTeamsOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
subscription CurrentTeams(
  $distinctOn: [currentTeamsSelectC
  $limit: Int,
  $offset: Int,
  $orderBy: [currentTeamsOrderBy!],
  $where: currentTeamsBoolExp
) {
  currentTeams(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
}
```

Name

Description

where -

filter the rows

`currentTeamsBoolExp` returned

}

VARIABLES

```
{  
  "distinctOn": ["abbreviation"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [currentTeamsOrderBy],  
  "where": currentTeamsBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "currentTeams": [  
      {  
        "abbreviation": "xyz789",  
        "classification": division,  
        "conference": "xyz789",  
        "conferenceId": smallint,  
        "division": "xyz789",  
        "school": "xyz789",  
        "teamId": 987  
      }  
    ]  
  }  
}
```

Subscriptions

currentTeamsAggregate

fetch aggregated fields from the table:
 "current_conferences"

Response

Returns a [currentTeamsAggregate!](#)

Arguments

Name	Description
distinctOn - [currentTeamsSelectColumns!]	distinct select on [currentTeamsSelectColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [currentTeamsOrderBy!]!	sort the rows by one or more [currentTeamsOrderBy!]! columns

QUERY

```
subscription CurrentTeamsAggregate(
  $distinctOn: [currentTeamsSelectC
  $limit: Int,
  $offset: Int,
  $orderBy: [currentTeamsOrderBy!],
  $where: currentTeamsBoolExp
) {
  currentTeamsAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...currentTeamsAvgFieldsFra
      }
      count
      max {
        ...currentTeamsMaxFieldsFra
      }
      min {
        ...currentTeamsMinFieldsFra
      }
    }
  }
}
```

Name	Description
where - <code>currentTeamsBoolExp</code>	filter the rows returned

```

    stdDev {
      ...currentTeamsStdDevFields
    }
    stdDevPop {
      ...currentTeamsStdDevPopFields
    }
    stdDevSamp {
      ...currentTeamsStdDevSampFields
    }
    sum {
      ...currentTeamsSumFields
    }
    varPop {
      ...currentTeamsVarPopFields
    }
    varSamp {
      ...currentTeamsVarSampFields
    }
    variance {
      ...currentTeamsVarianceFields
    }
  }
  nodes {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
}
}
}

```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
```

```
        "limit": 123,  
        "offset": 123,  
        "orderBy": [currentTeamsOrderBy],  
        "where": currentTeamsBoolExp  
    }  
}
```

RESPONSE

```
{  
  "data": {  
    "currentTeamsAggregate": {  
      "aggregate": currentTeamsAggr  
      "nodes": [currentTeams]  
    }  
  }  
}
```



Subscriptions

draftPicks

fetch data from the table: "draft_picks"

Response

QUERY

Returns `[DraftPicks!]!`

Arguments

Name	Description
<code>distinctOn - [DraftPicksSelectColumn]s</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [DraftPicksOrderBy!]</code>	sort the rows by one or more columns
<code>where - DraftPicksBoolExp</code>	filter the rows returned

```
subscription DraftPicks(
  $distinctOn: [DraftPicksSelectColumn]s
  $limit: Int,
  $offset: Int,
  $orderBy: [DraftPicksOrderBy!],
  $where: DraftPicksBoolExp
) {
  draftPicks(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    collegeAthleteRecord {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
    }
  }
}
```

```
        }
      positionId
      recruits {
        ...RecruitFragment
      }
      recruitsAggregate {
        ...RecruitAggregateFragment
      }
      teamId
      weight
    }
    collegeId
    collegeTeam {
      abbreviation
      active
      altColor
      altName
      classification
      color
      conference
      conferenceAbbreviation
      conferenceId
      conferenceShortName
      countryCode
      displayName
      division
      endYear
      id
      images
      mascot
      ncaaName
      nickname
      school
      shortDisplayName
      startYear
      twitter
    }
    collegeTeamId
    draftTeam {
      displayName
```

```
    id
    location
    logo
    mascot
    nickname
    picks {
      ...DraftPicksFragment
    }
    picksAggregate {
      ...DraftPicksAggregateFragm
    }
    shortDisplayName
  }
  grade
  height
  name
  nflTeamId
  overall
  overallRank
  pick
  position {
    abbreviation
    id
    name
  }
  positionId
  positionRank
  round
  weight
  year
}
}
```

VARIABLES

```
{
  "distinctOn": ["collegeId"],
  "limit": 987,
```

```
        "offset": 123,  
        "orderBy": [DraftPicksOrderBy],  
        "where": DraftPicksBoolExp  
    }  
}
```

RESPONSE

```
{  
  "data": {  
    "draftPicks": [  
      {  
        "collegeAthleteRecord": AthleteRecord,  
        "collegeId": 123,  
        "collegeTeam": historicalTeam,  
        "collegeTeamId": 987,  
        "draftTeam": DraftTeam,  
        "grade": smallint,  
        "height": smallint,  
        "name": "xyz789",  
        "nflTeamId": smallint,  
        "overall": smallint,  
        "overallRank": smallint,  
        "pick": smallint,  
        "position": DraftPosition,  
        "positionId": smallint,  
        "positionRank": smallint,  
        "round": smallint,  
        "weight": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}
```

Subscriptions

draftPosition

fetch data from the table:

"draft_position"

Response

Returns `[DraftPosition!]!`

Arguments

Name	Description
<code>distinctOn - [DraftPositionSelectColumns!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [DraftPositionOrderBy!]</code>	sort the rows by one or more columns

QUERY

```
subscription DraftPosition(  
  $distinctOn: [DraftPositionSelect  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [DraftPositionOrderBy!]!  
  $where: DraftPositionBoolExp  
) {  
  draftPosition(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
  ) {  
    abbreviation  
    id  
    name  
  }  
}
```

VARIABLES

Name	Description
where - <code>DraftPositionBoolExp</code>	filter the rows

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 123,
  "orderBy": [DraftPositionOrderBy]
  "where": DraftPositionBoolExp
}
```

RESPONSE

```
{
  "data": {
    "draftPosition": [
      {
        "abbreviation": "xyz789",
        "id": smallint,
        "name": "xyz789"
      }
    ]
  }
}
```

Subscriptions

draftTeam

fetch data from the table: "draft_team"

Response

Returns `[DraftTeam!]!`

Arguments

Name	Description
<code>distinctOn - [DraftTeamSelectColumns]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [DraftTeamOrderBy!]</code>	sort the rows by one or more columns
<code>where - DraftTeamBoolExp</code>	filter the rows returned

QUERY

```

subscription DraftTeam(
  $distinctOn: [DraftTeamSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [DraftTeamOrderBy!],
  $where: DraftTeamBoolExp
) {
  draftTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    displayName
    id
    location
    logo
    mascot
    nickname
    picks {
      collegeAthleteRecord {
        ...AthleteFragment
      }
    }
    collegeId
    collegeTeam {
      ...historicalTeamFragment
    }
    collegeTeamId
    draftTeam {
      ...DraftTeamFragment
    }
    grade
    height
    name
    nflTeamId
  }
}

```

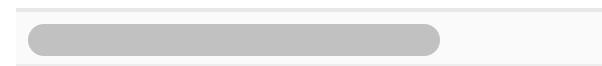
```
        overall
        overallRank
        pick
        position {
          ...DraftPositionFragment
        }
        positionId
        positionRank
        round
        weight
        year
      }
      picksAggregate {
        aggregate {
          ...DraftPicksAggregateField
        }
        nodes {
          ...DraftPicksFragment
        }
      }
      shortDisplayName
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["displayName"],
  "limit": 123,
  "offset": 123,
  "orderBy": [DraftTeamOrderBy],
  "where": DraftTeamBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "draftTeam": [  
      {  
        "displayName": "xyz789",  
        "id": smallint,  
        "location": "abc123",  
        "logo": "abc123",  
        "mascot": "abc123",  
        "nickname": "xyz789",  
        "picks": [DraftPicks],  
        "picksAggregate": DraftPick  
        "shortDisplayName": "xyz789"  
      }  
    ]  
  }  
}
```



Subscriptions

game

fetch data from the table: "game_info"

Response

Returns `[game!]!`

QUERY

```
subscription Game(  
  $distinctOn: [gameSelectColumn!],
```

Arguments

Name	Description
distinctOn - [gameSelectColumn!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [gameOrderBy!]	sort the rows by one or more columns
where - gameBoolExp	filter the rows returned

```
$limit: Int,
$offset: Int,
$orderBy: [gameOrderBy!],
$where: gameBoolExp
) {
  game(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    attendance
    awayClassification
    awayConference
    awayConferenceId
    awayConferenceInfo {
      abbreviation
      division
      id
      name
      shortName
      srName
    }
    awayEndElo
    awayLineScores
    awayPoints
    awayPostgameWinProb
    awayStartElo
    awayTeam
    awayTeamId
    awayTeamInfo {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
  }
}
```

```
conferenceGame
excitement
homeClassification
homeConference
homeConferenceId
homeConferenceInfo {
    abbreviation
    division
    id
    name
    shortName
    srName
}
homeEndElo
homeLineScores
homePoints
homePostgameWinProb
homeStartElo
homeTeam
homeTeamId
homeTeamInfo {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
}
id
lines {
    gameId
    linesProviderId
    moneylineAway
    moneylineHome
    overUnder
    overUnderOpen
    provider {
        ...LinesProviderFragment
    }
}
```

```
        spread
        spreadOpen
    }
    linesAggregate {
        aggregate {
            ...GameLinesAggregateFields
        }
        nodes {
            ...GameLinesFragment
        }
    }
    mediaInfo {
        mediaType
        name
    }
    neutralSite
    notes
    season
    seasonType
    startDate
    startTimeTbd
    status
    venueId
    weather {
        condition {
            ...WeatherConditionFragment
        }
        dewpoint
        gameId
        humidity
        precipitation
        pressure
        snowfall
        temperature
        weatherConditionCode
        windDirection
        windGust
        windSpeed
    }
    week
```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["attendance"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [gameOrderBy],  
  "where": gameBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "game": [  
      {  
        "attendance": 987,  
        "awayClassification": divis  
        "awayConference": "abc123",  
        "awayConferenceId": smallint  
        "awayConferenceInfo": Confe  
        "awayEndElo": 123,  
        "awayLineScores": [smallint  
        "awayPoints": smallint,  
        "awayPostgameWinProb": nume  
        "awayStartElo": 987,  
        "awayTeam": "xyz789",  
        "awayTeamId": 123,  
        "awayTeamInfo": currentTeam  
        "conferenceGame": true,  
        "excitement": numeric,  
        "homeClassification": divis  
        "homeConference": "abc123",  
        "homeTeam": "xyz789",  
        "homeTeamId": 123,  
        "homeTeamInfo": currentTeam  
      }  
    ]  
  }  
}
```

```
        "homeConferenceId": smallint,
        "homeConferenceInfo": Conference,
        "homeEndElo": 123,
        "homeLineScores": [smallint],
        "homePoints": smallint,
        "homePostgameWinProb": numeric,
        "homeStartElo": 987,
        "homeTeam": "xyz789",
        "homeTeamId": 123,
        "homeTeamInfo": currentTeam,
        "id": 123,
        "lines": [GameLines],
        "linesAggregate": GameLines,
        "mediaInfo": [GameMedia],
        "neutralSite": false,
        "notes": "xyz789",
        "season": smallint,
        "seasonType": season_type,
        "startDate": timestamp,
        "startTimeTbd": false,
        "status": game_status,
        "venueId": 123,
        "weather": GameWeather,
        "week": smallint
    }
]
}
```

Subscriptions

gameAggregate

fetch aggregated fields from the table:
 "game_info"

Response

Returns a [gameAggregate!](#)

Arguments

Name	Description
distinctOn - [gameSelectColumn!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [gameOrderBy!]	sort the rows by one or more columns
where - gameBoolExp	filter the rows returned

QUERY

```
subscription GameAggregate(
  $distinctOn: [gameSelectColumn!]!,
  $limit: Int,
  $offset: Int,
  $orderBy: [gameOrderBy!]!,
  $where: gameBoolExp
) {
  gameAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...gameAvgFieldsFragment
      }
      count
      max {
        ...gameMaxFieldsFragment
      }
      min {
        ...gameMinFieldsFragment
      }
      stddev {
        ...gameStddevFieldsFragment
      }
      stddevPop {
        ...gameStddevPopFieldsFragm
      }
      stddevSamp {
        ...gameStddevSampFieldsFrag
      }
    }
  }
}
```

```
        }
      sum {
        ...gameSumFieldsFragment
      }
      varPop {
        ...gameVarPopFieldsFragment
      }
      varSamp {
        ...gameVarSampFieldsFragmen
      }
      variance {
        ...gameVarianceFieldsFragme
      }
    }
  nodes {
    attendance
    awayClassification
    awayConference
    awayConferenceId
    awayConferenceInfo {
      ...ConferenceFragment
    }
    awayEndElo
    awayLineScores
    awayPoints
    awayPostgameWinProb
    awayStartElo
    awayTeam
    awayTeamId
    awayTeamInfo {
      ...currentTeamsFragment
    }
    conferenceGame
    excitement
    homeClassification
    homeConference
    homeConferenceId
    homeConferenceInfo {
      ...ConferenceFragment
    }
  }
}
```

```
    homeEndElo
    homeLineScores
    homePoints
    homePostgameWinProb
    homeStartElo
    homeTeam
    homeTeamId
    homeTeamInfo {
        ...currentTeamsFragment
    }
    id
    lines {
        ...GameLinesFragment
    }
    linesAggregate {
        ...GameLinesAggregateFragme
    }
    mediaInfo {
        ...GameMediaFragment
    }
    neutralSite
    notes
    season
    seasonType
    startDate
    startTimeTbd
    status
    venueId
    weather {
        ...GameWeatherFragment
    }
    week
}
}
```

VARIABLES

```
{  
  "distinctOn": ["attendance"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [gameOrderBy],  
  "where": gameBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "gameAggregate": {  
      "aggregate": gameAggregateFie  
      "nodes": [game]  
    }  
  }  
}
```



Subscriptions

gameLines

fetch data from the table: "game_lines"

Response**QUERY**

Returns [GameLines!]!

Arguments

Name	Description
distinctOn - [GameLinesSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GameLinesOrderBy!]	sort the rows by one or more columns
where - GameLinesBoolExp	filter the rows returned

```

subscription GameLines(
  $distinctOn: [GameLinesSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [GameLinesOrderBy!],
  $where: GameLinesBoolExp
) {
  gameLines(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    gameId
    linesProviderId
    moneylineAway
    moneylineHome
    overUnder
    overUnderOpen
    provider {
      id
      name
    }
    spread
    spreadOpen
  }
}

```

VARIABLES

```
{
  "distinctOn": ["gameId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [GameLinesOrderBy],
```

```
        "where": GameLinesBoolExp
    }
```

RESPONSE

```
{
  "data": {
    "gameLines": [
      {
        "gameId": 123,
        "linesProviderId": 987,
        "moneylineAway": 123,
        "moneylineHome": 987,
        "overUnder": numeric,
        "overUnderOpen": numeric,
        "provider": LinesProvider,
        "spread": numeric,
        "spreadOpen": numeric
      }
    ]
  }
}
```

Subscriptions

gameLinesAggregate

fetch aggregated fields from the table:
"game_lines"

Response

Returns a [GameLinesAggregate!](#)

Arguments

Name	Description
distinctOn - [GameLinesSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GameLinesOrderBy!]	sort the rows by one or more columns
where - GameLinesBoolExp	filter the rows returned

QUERY

```
subscription GameLinesAggregate(
  $distinctOn: [GameLinesSelectColu
  $limit: Int,
  $offset: Int,
  $orderBy: [GameLinesOrderBy!],
  $where: GameLinesBoolExp
) {
  gameLinesAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...GameLinesAvgFieldsFragme
      }
      count
      max {
        ...GameLinesMaxFieldsFragme
      }
      min {
        ...GameLinesMinFieldsFragme
      }
      stddev {
        ...GameLinesStddevFieldsFra
      }
      stddevPop {
        ...GameLinesStddevPopFields
      }
      stddevSamp {
        ...GameLinesStddevSampField
      }
      sum {
        ...GameLinesSumFieldsFragme
      }
    }
  }
}
```

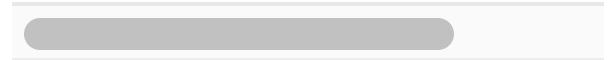
```
        }
      varPop {
        ...GameLinesVarPopFieldsFragment
      }
      varSamp {
        ...GameLinesVarSampFieldsFragment
      }
      variance {
        ...GameLinesVarianceFieldsFragment
      }
    }
  nodes {
    gameId
    linesProviderId
    moneylineAway
    moneylineHome
    overUnder
    overUnderOpen
    provider {
      ...LinesProviderFragment
    }
    spread
    spreadOpen
  }
}
```

VARIABLES

```
{
  "distinctOn": ["gameId"],
  "limit": 987,
  "offset": 987,
  "orderBy": [GameLinesOrderBy],
  "where": GameLinesBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "gameLinesAggregate": {  
      "aggregate": GameLinesAggregate  
      "nodes": [GameLines]  
    }  
  }  
}
```



Subscriptions

gameMedia

fetch data from the table: "game_media"

Response**QUERY**

Returns `[GameMedia!]!`

```
subscription GameMedia(  
  $distinctOn: [GameMediaSelectColu  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [GameMediaOrderBy!],  
  $where: GameMediaBoolExp  
) {  
  gameMedia(
```

Arguments

Name	Description
distinctOn - [GameMediaSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GameMediaOrderBy!]	sort the rows by one or more columns
where - GameMediaBoolExp	filter the rows returned

```
distinctOn: $distinctOn,
limit: $limit,
offset: $offset,
orderBy: $orderBy,
where: $where
) {
  mediaType
  name
}
```

VARIABLES

```
{
  "distinctOn": ["mediaType"],
  "limit": 123,
  "offset": 987,
  "orderBy": [GameMediaOrderBy],
  "where": GameMediaBoolExp
}
```

RESPONSE

```
{
  "data": {
    "gameMedia": [
      {
        "mediaType": media_type,
        "name": "xyz789"
      }
    ]
  }
}
```

Subscriptions

gamePlayerStat

fetch data from the table:

"game_player_stat"

Response

Returns `[GamePlayerStat!]!`

Arguments

Name	Description
<code>distinctOn - [GamePlayerStatSelectColumn!]!</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [GamePlayerStatOrderBy!]!</code>	sort the rows by one or more columns

QUERY

```

subscription GamePlayerStat(
  $distinctOn: [GamePlayerStatSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [GamePlayerStatOrderBy!]!
  $where: GamePlayerStatBoolExp
) {
  gamePlayerStat(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...
      }
    }
  }
}

```

Name	Description	
where - GamePlayerStatBool	filter the rows returned	...AthleteTeamAggregateFrag } firstName height hometown { ...HometownFragment } hometownId id jersey lastName name position { ...PositionFragment } positionId recruits { ...RecruitFragment } recruitsAggregate { ...RecruitAggregateFragment } teamId weight } athleteId gameTeam { endElo game { ...gameFragment } gameId gamePlayerStats { ...GamePlayerStatFragment } gamePlayerStatsAggregate { ...GamePlayerStatAggregateF } homeAway lineScores

```
        points
        startElo
        teamId
        winProb
    }
    gameTeamId
    id
    playerStatCategory {
        gamePlayerStats {
            ...GamePlayerStatFragment
        }
        gamePlayerStatsAggregate {
            ...GamePlayerStatAggregateF
        }
        name
    }
    playerStatType {
        name
    }
    stat
}
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 123,
  "offset": 123,
  "orderBy": [GamePlayerStatOrderBy],
  "where": GamePlayerStatBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "gamePlayerStat": [  
      {  
        "athlete": Athlete,  
        "athleteId": bigint,  
        "gameTeam": GameTeam,  
        "gameTeamId": bigint,  
        "id": bigint,  
        "playerStatCategory": Playe  
        "playerStatType": PlayerSta  
        "stat": "abc123"  
      }  
    ]  
  }  
}
```

Subscriptions

gamePlayerStatAggregate

fetch aggregated fields from the table:

"game_player_stat"

Response

Returns a [GamePlayerStatAggregate!](#)

QUERY

```
subscription GamePlayerStatAggregat  
$distinctOn: [GamePlayerStatSelec
```

Arguments

Name	Description
distinctOn - [GamePlayerStatSelectColumn!]!	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GamePlayerStatOrderBy!]!	sort the rows by one or more columns
where - GamePlayerStatBoolExp	filter the rows

```
$limit: Int,
$offset: Int,
$orderBy: [GamePlayerStatOrderBy!]!
$where: GamePlayerStatBoolExp
) {
  gamePlayerStatAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...GamePlayerStatAvgFieldsF
      }
      count
      max {
        ...GamePlayerStatMaxFieldsF
      }
      min {
        ...GamePlayerStatMinFieldsF
      }
      stdDev {
        ...GamePlayerStatStddevFiel
      }
      stdDevPop {
        ...GamePlayerStatStddevPopF
      }
      stdDevSamp {
        ...GamePlayerStatStddevSamp
      }
      sum {
        ...GamePlayerStatSumFieldsF
      }
      varPop {
        ...GamePlayerStatVarPopFiel
      }
      varSamp {
        ...GamePlayerStatVarSampFie
      }
    }
  }
}
```

```
        }
      variance {
        ...GamePlayerStatVarianceFi
      }
    }
  nodes {
    athlete {
      ...AthleteFragment
    }
    athleteId
    gameTeam {
      ...GameTeamFragment
    }
    gameTeamId
    id
    playerStatCategory {
      ...PlayerStatCategoryFragme
    }
    playerStatType {
      ...PlayerStatTypeFragment
    }
    stat
  }
}
```

VARIABLES

```
{
  "distinctOn": ["athleteId"],
  "limit": 123,
  "offset": 987,
  "orderBy": [GamePlayerStatOrderBy],
  "where": GamePlayerStatBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "gamePlayerStatAggregate": {  
      "aggregate": GamePlayerStatAg  
      "nodes": [GamePlayerStat]  
    }  
  }  
}
```



Subscriptions

gameTeam

fetch data from the table: "game_team"

Response

Returns [\[GameTeam!\]!](#)

Arguments**QUERY**

```
subscription GameTeam(  
  $distinctOn: [GameTeamSelectColumn]  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [GameTeamOrderBy!],  
  $where: GameTeamBoolExp  
) {  
  gameTeam(  
    $distinctOn: [GameTeamSelectColumn]  
    $limit: Int,  
    $offset: Int,  
    $orderBy: [GameTeamOrderBy!],  
    $where: GameTeamBoolExp  
  )  
}
```

Name	Description
distinctOn - [GameTeamSelectColumn]	distinct select on [GameTeamSelectColumn]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GameTeamOrderBy!]	sort the rows by one or more columns
where - GameTeamBoolExp	filter the rows returned

```

distinctOn: $distinctOn,
limit: $limit,
offset: $offset,
orderBy: $orderBy,
where: $where
) {
endElo
game {
attendance
awayClassification
awayConference
awayConferenceId
awayConferenceInfo {
...ConferenceFragment
}
awayEndElo
awayLineScores
awayPoints
awayPostgameWinProb
awayStartElo
awayTeam
awayTeamId
awayTeamInfo {
...currentTeamsFragment
}
conferenceGame
excitement
homeClassification
homeConference
homeConferenceId
homeConferenceInfo {
...ConferenceFragment
}
homeEndElo
homeLineScores
homePoints
homePostgameWinProb
homeStartElo
homeTeam
homeTeamId

```

```
homeTeamInfo {  
    ...currentTeamsFragment  
}  
id  
lines {  
    ...GameLinesFragment  
}  
linesAggregate {  
    ...GameLinesAggregateFragme  
}  
mediaInfo {  
    ...GameMediaFragment  
}  
neutralSite  
notes  
season  
seasonType  
startDate  
startTimeTbd  
status  
venueId  
weather {  
    ...GameWeatherFragment  
}  
week  
}  
gameId  
gamePlayerStats {  
    athlete {  
        ...AthleteFragment  
    }  
    athleteId  
    gameTeam {  
        ...GameTeamFragment  
    }  
    gameTeamId  
    id  
    playerStatCategory {  
        ...PlayerStatCategoryFragme  
    }  
}
```

```

    playerStatType {
      ...PlayerStatTypeFragment
    }
    stat
  }
  gamePlayerStatsAggregate {
    aggregate {
      ...GamePlayerStatAggregateFragment
    }
    nodes {
      ...GamePlayerStatFragment
    }
  }
  homeAway
  lineScores
  points
  startElo
  teamId
  winProb
}
}

```

VARIABLES

```
{
  "distinctOn": ["endElo"],
  "limit": 123,
  "offset": 123,
  "orderBy": [GameTeamOrderBy],
  "where": GameTeamBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```
        "gameTeam": [  
          {  
            "endElo": 123,  
            "game": game,  
            "gameId": 123,  
            "gamePlayerStats": [GamePla  
            "gamePlayerStatsAggregate":  
              "homeAway": home_away,  
              "lineScores": [smallint],  
              "points": smallint,  
              "startElo": 987,  
              "teamId": 987,  
              "winProb": numeric  
            }  
          ]  
        }  
      }
```

Subscriptions

gameWeather

fetch data from the table:

"game_weather"

Response

Returns [GameWeather!]!

QUERY

```
subscription GameWeather(  
  $distinctOn: [GameWeatherSelectCo
```

Arguments

Name	Description
distinctOn - [GameWeatherSelectColumns]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [GameWeatherOrderBy!]	sort the rows by one or more columns
where - GameWeatherBoolExp	filter the rows

```

$limit: Int,
$offset: Int,
$orderBy: [GameWeatherOrderBy!],
$where: GameWeatherBoolExp
) {
  gameWeather(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    condition {
      description
      id
    }
    dewpoint
    gameId
    humidity
    precipitation
    pressure
    snowfall
    temperature
    weatherConditionCode
    windDirection
    windGust
    windSpeed
  }
}

```

VARIABLES

```
{
  "distinctOn": ["dewpoint"],
  "limit": 123,
  "offset": 987,
  "orderBy": [GameWeatherOrderBy],
```

```
        "where": GameWeatherBoolExp
```

```
}
```

RESPONSE

```
{
  "data": {
    "gameWeather": [
      {
        "condition": WeatherCondition,
        "dewpoint": numeric,
        "gameId": 123,
        "humidity": numeric,
        "precipitation": numeric,
        "pressure": numeric,
        "snowfall": numeric,
        "temperature": numeric,
        "weatherConditionCode": smallInt,
        "windDirection": numeric,
        "windGust": numeric,
        "windSpeed": numeric
      }
    ]
  }
}
```



Subscriptions

historicalTeam

fetch data from the table: "team_info"

Response

Returns `[historicalTeam!]!`

Arguments

Name	Description
<code>distinctOn - [historicalTeamSelectColumn!]!</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [historicalTeamOrderBy!]!</code>	sort the rows by one or more columns
<code>where - historicalTeamBoolExp</code>	filter the rows

QUERY

```

subscription HistoricalTeam(
  $distinctOn: [historicalTeamSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [historicalTeamOrderBy!]!
  $where: historicalTeamBoolExp
) {
  historicalTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    active
    altColor
    altName
    classification
    color
    conference
    conferenceAbbreviation
    conferenceId
    conferenceShortName
    countryCode
    displayName
    division
    endYear
    id
  }
}

```

```
    images
    mascot
    ncaaName
    nickname
    school
    shortDisplayName
    startYear
    twitter
  }
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 123,
  "orderBy": [historicalTeamOrderBy],
  "where": historicalTeamBoolExp
}
```

RESPONSE

```
{
  "data": {
    "historicalTeam": [
      {
        "abbreviation": "xyz789",
        "active": true,
        "altColor": "xyz789",
        "altName": "xyz789",
        "classification": division,
        "color": "abc123",
        "conference": "abc123",
        "conferenceAbbreviation": "xyz789"
      }
    ]
  }
}
```

```
"conferenceId": smallint,  
"conferenceShortName": "xyz  
"countryCode": "abc123",  
"displayName": "xyz789",  
"division": "abc123",  
"endYear": smallint,  
"id": 123,  
"images": ["xyz789"],  
"mascot": "abc123",  
"ncaaName": "abc123",  
"nickname": "xyz789",  
"school": "xyz789",  
"shortDisplayName": "xyz789",  
"startYear": smallint,  
"twitter": "xyz789"  
}  
]  
}  
}
```

Subscriptions

historicalTeamAggregate

fetch aggregated fields from the table:
"team_info"

Response

QUERY

Returns a [historicalTeamAggregate](#)!

Arguments

Name	Description
distinctOn - [historicalTeamSelectionColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [historicalTeamOrderBy!]	sort the rows by one or more columns
where - historicalTeamBoolExp	filter the rows

```
subscription HistoricalTeamAggregate {
  $distinctOn: [historicalTeamSelectionColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [historicalTeamOrderBy!]!
  $where: historicalTeamBoolExp
} {
  historicalTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
) {
  aggregate {
    avg {
      ...historicalTeamAvgFieldsF
    }
    count
    max {
      ...historicalTeamMaxFieldsF
    }
    min {
      ...historicalTeamMinFieldsF
    }
    stdDev {
      ...historicalTeamStddevFiel
    }
    stdDevPop {
      ...historicalTeamStddevPopF
    }
    stdDevSamp {
      ...historicalTeamStddevSamp
    }
    sum {
      ...historicalTeamSumFieldsF
    }
    varPop {
      ...historicalTeamVarPopFiel
    }
  }
}
```

```
        }
      varSamp {
        ...historicalTeamVarSampFi
      }
      variance {
        ...historicalTeamVarianceFi
      }
    }
    nodes {
      abbreviation
      active
      altColor
      altName
      classification
      color
      conference
      conferenceAbbreviation
      conferenceId
      conferenceShortName
      countryCode
      displayName
      division
      endYear
      id
      images
      mascot
      ncaaName
      nickname
      school
      shortDisplayName
      startYear
      twitter
    }
  }
}
```

VARIABLES

```
{  
  "distinctOn": ["abbreviation"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [historicalTeamOrderBy  
  "where": historicalTeamBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "historicalTeamAggregate": {  
      "aggregate": historicalTeamAg  
      "nodes": [historicalTeam]  
    }  
  }  
}
```

Subscriptions

hometown

fetch data from the table: "hometown"

Response**Returns [Hometown!]!****Arguments**

Name	Description
distinctOn - [HometownSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [HometownOrderBy!]	sort the rows by one or more columns
where - HometownBoolExp	filter the rows returned

QUERY

```

subscription Hometown(
  $distinctOn: [HometownSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [HometownOrderBy!]!,
  $where: HometownBoolExp
) {
  hometown(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athletes {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregateFragment
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
    }
  }
}

```

```
position {  
  ...PositionFragment  
}  
positionId  
recruits {  
  ...RecruitFragment  
}  
recruitsAggregate {  
  ...RecruitAggregateFragment  
}  
teamId  
weight  
}  
athletesAggregate {  
  aggregate {  
    ...AthleteAggregateFieldsFr  
  }  
  nodes {  
    ...AthleteFragment  
  }  
}  
city  
country  
countyFips  
latitude  
longitude  
recruits {  
  athlete {  
    ...AthleteFragment  
  }  
  college {  
    ...currentTeamsFragment  
  }  
  height  
  hometown {  
    ...HometownFragment  
  }  
  id  
  name  
  overallRank
```

```
        position {
          ...RecruitPositionFragment
        }
        positionRank
        ranking
        rating
        recruitSchool {
          ...RecruitSchoolFragment
        }
        recruitType
        stars
        weight
        year
      }
      recruitsAggregate {
        aggregate {
          ...RecruitAggregateFieldsFr
        }
        nodes {
          ...RecruitFragment
        }
      }
      state
    }
  }
```

VARIABLES

```
{
  "distinctOn": ["city"],
  "limit": 123,
  "offset": 987,
  "orderBy": [HometownOrderBy],
  "where": HometownBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "hometown": [  
      {  
        "athletes": [Athlete],  
        "athletesAggregate": Athlet  
        "city": "abc123",  
        "country": "abc123",  
        "countyFips": "abc123",  
        "latitude": numeric,  
        "longitude": numeric,  
        "recruits": [Recruit],  
        "recruitsAggregate": Recrui  
        "state": "abc123"  
      }  
    ]  
  }  
}
```

Subscriptions

hometownAggregate

fetch aggregated fields from the table:
"hometown"

Response

Returns a [HometownAggregate!](#)

Arguments

Name	Description
distinctOn - [HometownSelectColumns]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [HometownOrderBy!]	sort the rows by one or more columns
where - HometownBoolExp	filter the rows returned

QUERY

```
subscription HometownAggregate(
  $distinctOn: [HometownSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [HometownOrderBy!]!,
  $where: HometownBoolExp
) {
  hometownAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...HometownAvgFieldsFragment
      }
      count
      max {
        ...HometownMaxFieldsFragment
      }
      min {
        ...HometownMinFieldsFragment
      }
      stdDev {
        ...HometownStddevFieldsFragment
      }
      stdDevPop {
        ...HometownStddevPopFieldsFragment
      }
      stdDevSamp {
        ...HometownStddevSampFieldsFragment
      }
      sum {
        ...HometownSumFieldsFragment
      }
    }
  }
}
```

```
    varPop {  
      ...HometownVarPopFieldsFrag  
    }  
    varSamp {  
      ...HometownVarSampFieldsFra  
    }  
    variance {  
      ...HometownVarianceFieldsFr  
    }  
  }  
  nodes {  
    athletes {  
      ...AthleteFragment  
    }  
    athletesAggregate {  
      ...AthleteAggregateFragment  
    }  
    city  
    country  
    countyFips  
    latitude  
    longitude  
    recruits {  
      ...RecruitFragment  
    }  
    recruitsAggregate {  
      ...RecruitAggregateFragment  
    }  
    state  
  }  
}
```

VARIABLES

```
{  
  "distinctOn": ["city"],  
  "limit": 123,
```

```
"offset": 987,  
"orderBy": [HometownOrderBy],  
"where": HometownBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "hometownAggregate": {  
      "aggregate": HometownAggregate,  
      "nodes": [Hometown]  
    }  
  }  
}
```



Subscriptions

linesProvider

fetch data from the table:

"lines_provider"

Response

QUERY

Returns **[LinesProvider!]!**

Arguments

Name	Description
distinctOn - [LinesProviderSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [LinesProviderOrderBy!]	sort the rows by one or more columns
where - LinesProviderBoolExp	filter the rows

```
subscription LinesProvider(
  $distinctOn: [LinesProviderSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [LinesProviderOrderBy!]!
  $where: LinesProviderBoolExp
) {
  linesProvider(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    name
  }
}
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 987,
  "offset": 123,
  "orderBy": [LinesProviderOrderBy!]
  "where": LinesProviderBoolExp
}
```

RESPONSE

```
{
  "data": {
    "linesProvider": [

```

```
{
  "id": 987,
  "name": "xyz789"
}
]
```

Subscriptions

linesProviderAggregate

fetch aggregated fields from the table:

"lines_provider"

Response

Returns a [LinesProviderAggregate!](#)

Arguments

Name	Description
<code>distinctOn - [LinesProviderSelectColumns!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned

QUERY

```
subscription LinesProviderAggregate {
  $distinctOn: [LinesProviderSelect
    $limit: Int,
    $offset: Int,
    $orderBy: [LinesProviderOrderBy!]
    $where: LinesProviderBoolExp
  ) {
    linesProviderAggregate(
      distinctOn: $distinctOn,
      limit: $limit,
      offset: $offset,
      orderBy: $orderBy,
      where: $where
    ) {
      aggregate {
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[LinesProviderOrderBy!]</code>	sort the rows by one or more columns
where - <code>LinesProviderBoolExpr!</code>	filter the rows

```

avg {
  ...LinesProviderAvgFieldsFr
}
count
max {
  ...LinesProviderMaxFieldsFr
}
min {
  ...LinesProviderMinFieldsFr
}
stddev {
  ...LinesProviderStddevField
}
stddevPop {
  ...LinesProviderStddevPopFi
}
stddevSamp {
  ...LinesProviderStddevSampF
}
sum {
  ...LinesProviderSumFieldsFr
}
varPop {
  ...LinesProviderVarPopField
}
varSamp {
  ...LinesProviderVarSampFiel
}
variance {
  ...LinesProviderVarianceFie
}
}
nodes {
  id
  name
}

```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 123,  
  "offset": 987,  
  "orderBy": [LinesProviderOrderBy]  
  "where": LinesProviderBoolExp  
}
```



RESPONSE

```
{  
  "data": {  
    "linesProviderAggregate": {  
      "aggregate": LinesProviderAgg  
      "nodes": [LinesProvider]  
    }  
  }  
}
```



Subscriptions

playerStatCategory

fetch data from the table:

"player_stat_category"

Response

Returns [PlayerStatCategory!]!

Arguments

Name	Description
distinctOn - [PlayerStatCategoryColumns!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatCategoryOrderBy!]	sort the rows by one or more columns
where - PlayerStatCategoryFilter	filter the rows

QUERY

```
subscription PlayerStatCategory(
  $distinctOn: [PlayerStatCategory$columns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PlayerStatCategoryOrder$by!]!
  $where: PlayerStatCategoryBoolExp
) {
  playerStatCategory(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    gamePlayerStats {
      athlete {
        ...AthleteFragment
      }
      athleteId
      gameTeam {
        ...GameTeamFragment
      }
      gameId
      playerStatCategory {
        ...PlayerStatCategoryFramme
      }
      playerStatType {
        ...
      }
    }
  }
}
```

```
        ...PlayerStatTypeFragment
    }
    stat
}
gamePlayerStatsAggregate {
    aggregate {
        ...GamePlayerStatAggregateF
    }
    nodes {
        ...GamePlayerStatFragment
    }
}
name
}
```

VARIABLES

```
{
  "distinctOn": ["name"],
  "limit": 123,
  "offset": 123,
  "orderBy": [PlayerStatCategoryOrd
  "where": PlayerStatCategoryBoolEx
}
```

RESPONSE

```
{
  "data": {
    "playerStatCategory": [
      {
        "gamePlayerStats": [GamePla
        "gamePlayerStatsAggregate": [
          {
            "name": "xyz789"
          }
        ]
      }
    ]
  }
}
```

```
        }  
    ]  
}  
}
```

Subscriptions

playerStatCategoryAggregate

fetch aggregated fields from the table:
"player_stat_category"

Response

Returns a
[PlayerStatCategoryAggregate!](#)

Arguments

Name	Description
<code>distinctOn - [PlayerStatCategoryColumns!]</code>	distinct select on [PlayerStatCategoryColumns!]
<code>limit - Int</code>	limit the number of rows returned

QUERY

```
subscription PlayerStatCategoryAggr  
$distinctOn: [PlayerStatCategoryS  
$limit: Int,  
$offset: Int,  
$orderBy: [PlayerStatCategoryOrde  
$where: PlayerStatCategoryBoolExp  
) {  
playerStatCategoryAggregate(  
distinctOn: $distinctOn,  
limit: $limit,  
offset: $offset,  
orderBy: $orderBy,  
where: $where  
) {
```

Name	Description
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatCategoryOrderBy!]	sort the rows by one or more columns
where - PlayerStatCategoryBoolean	filter the rows

```

aggregate {
  count
  max {
    ...PlayerStatCategoryMaxField
  }
  min {
    ...PlayerStatCategoryMinField
  }
}
nodes {
  gamePlayerStats {
    ...GamePlayerStatFragment
  }
  gamePlayerStatsAggregate {
    ...GamePlayerStatAggregateField
  }
  name
}
}
}
}

```

VARIABLES

```
{
  "distinctOn": ["name"],
  "limit": 987,
  "offset": 987,
  "orderBy": [PlayerStatCategoryOrderBy],
  "where": PlayerStatCategoryBoolean
}
```



RESPONSE

```
{
  "data": {
```

```

    "playerStatCategoryAggregate":  

      "aggregate": PlayerStatCatego  

      "nodes": [PlayerStatCategory]  

    }  

  }  

}

```

Subscriptions

playerStatType

fetch data from the table:

"player_stat_type"

Response

Returns `[PlayerStatType!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[PlayerStatTypeSelectionColumn!]!</code>	distinct select on

QUERY

```

subscription PlayerStatType(  

  $distinctOn: [PlayerStatTypeSelec  

  $limit: Int,  

  $offset: Int,  

  $orderBy: [PlayerStatTypeOrderBy!  

  $where: PlayerStatTypeBoolExp  

) {  

  playerStatType(  

    distinctOn: $distinctOn,  

    limit: $limit,  

    offset: $offset,

```

Name	Description
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatTypeOrderBy!]	sort the rows by one or more columns
where - PlayerStatTypeBoolExp	filter the rows

```

    orderBy: $orderBy,
    where: $where
  ) {
    name
  }
}

```

VARIABLES

```
{
  "distinctOn": ["name"],
  "limit": 123,
  "offset": 123,
  "orderBy": [PlayerStatTypeOrderBy],
  "where": PlayerStatTypeBoolExp
}
```

RESPONSE

```
{
  "data": {
    "playerStatType": [{"name": "ab"}]
  }
}
```

Subscriptions

playerStatTypeAggregate

fetch aggregated fields from the table:
 "player_stat_type"

Response

Returns a [PlayerStatTypeAggregate!](#)

Arguments

Name	Description
distinctOn - [PlayerStatTypeSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PlayerStatTypeOrderBy!]	sort the rows by one or more columns
where - PlayerStatTypeBoolExp	filter the rows

QUERY

```
subscription PlayerStatTypeAggregate {
  $distinctOn: [PlayerStatTypeSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PlayerStatTypeOrderBy!]!
  $where: PlayerStatTypeBoolExp
} {
  playerStatTypeAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      count
      max {
        ...PlayerStatTypeMaxFieldsF
      }
      min {
        ...PlayerStatTypeMinFieldsF
      }
    }
    nodes {
      name
    }
  }
}
```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["name"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [PlayerStatTypeOrderBy  
  "where": PlayerStatTypeBoolExp  
}
```



RESPONSE

```
{  
  "data": {  
    "playerStatTypeAggregate": {  
      "aggregate": PlayerStatTypeAg  
      "nodes": [PlayerStatType]  
    }  
  }  
}
```



Subscriptions

poll

fetch data from the table: "poll"

Response

Returns `[Poll!]!`

Arguments

Name	Description
<code>distinctOn - [PollSelectColumn!]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [PollOrderBy!]</code>	sort the rows by one or more columns
<code>where - PollBoolExp</code>	filter the rows returned

QUERY

```
subscription Poll(
  $distinctOn: [PollSelectColumn!],
  $limit: Int,
  $offset: Int,
  $orderBy: [PollOrderBy!],
  $where: PollBoolExp
) {
  poll(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    pollType {
      abbreviation
      id
      name
      polls {
        ...PollFragment
      }
      shortName
    }
    rankings {
      firstPlaceVotes
      points
      poll {
        ...PollFragment
      }
    }
  }
}
```

```
rank
team {
  ...currentTeamsFragment
}
}
season
seasonType
week
}
```

VARIABLES

```
{
  "distinctOn": ["season"],
  "limit": 987,
  "offset": 123,
  "orderBy": [PollOrderBy],
  "where": PollBoolExp
}
```

RESPONSE

```
{
  "data": {
    "poll": [
      {
        "pollType": PollType,
        "rankings": [PollRank],
        "season": 987,
        "seasonType": season_type,
        "week": smallint
      }
    ]
  }
}
```

```

    }
}
}
```

Subscriptions

pollRank

fetch data from the table: "poll_rank"

Response

Returns `[PollRank!]!`

Arguments

Name	Description
<code>distinctOn - [PollRankSelectColumns]</code>	distinct select on columns
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>

QUERY

```

subscription PollRank(
  $distinctOn: [PollRankSelectColumn!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [PollRankOrderBy!]!,
  $where: PollRankBoolExp
) {
  pollRank(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    firstPlaceVotes
    points
    poll {
      pollType {
```

Name	Description
orderBy - [PollRankOrderBy!]	sort the rows by one or more columns
where - PollRankBoolExp	filter the rows returned

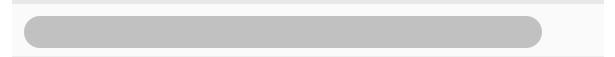
```
        ...PollTypeFragment  
    }  
    rankings {  
        ...PollRankFragment  
    }  
    season  
    seasonType  
    week  
}  
rank  
team {  
    abbreviation  
    classification  
    conference  
    conferenceId  
    division  
    school  
    teamId  
}
```

VARIABLES

```
{  
  "distinctOn": ["firstPlaceVotes"]  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [PollRankOrderBy],  
  "where": PollRankBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "pollRank": [  
      {  
        "firstPlaceVotes": smallint  
        "points": 987,  
        "poll": Poll,  
        "rank": smallint,  
        "team": currentTeams  
      }  
    ]  
  }  
}
```



Subscriptions

pollType

fetch data from the table: "poll_type"

Response

Returns [PollType!]!

Arguments

QUERY

```
subscription PollType(  
  $distinctOn: [PollTypeSelectColumn  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [PollTypeOrderBy!],  
  $where: PollTypeBoolExp  
) {
```

Name	Description
distinctOn - [PollTypeSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PollTypeOrderBy!]	sort the rows by one or more columns
where - PollTypeBoolExp	filter the rows returned

```

pollType(
  distinctOn: $distinctOn,
  limit: $limit,
  offset: $offset,
  orderBy: $orderBy,
  where: $where
) {
  abbreviation
  id
  name
  polls {
    pollType {
      ...PollTypeFragment
    }
    rankings {
      ...PollRankFragment
    }
    season
    seasonType
    week
  }
  shortName
}
}

```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 123,
  "offset": 987,
  "orderBy": [PollTypeOrderBy!],
  "where": PollTypeBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "pollType": [  
      {  
        "abbreviation": "xyz789",  
        "id": 123,  
        "name": "xyz789",  
        "polls": [Poll],  
        "shortName": "xyz789"  
      }  
    ]  
  }  
}
```

Subscriptions

position

fetch data from the table: "position"

Response

Returns **[Position!]!**

Arguments**QUERY**

```
subscription Position(  
  $distinctOn: [PositionSelectColumn  
  $limit: Int,  
  $offset: Int,
```

Name	Description
distinctOn - [PositionSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [PositionOrderBy!]	sort the rows by one or more columns
where - PositionBoolExp	filter the rows returned

```
$orderBy: [PositionOrderBy!],
$where: PositionBoolExp
) {
  position(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    abbreviation
    athletes {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
      positionId
      recruits {
        ...RecruitFragment
      }
    }
  }
}
```

```
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
    teamId
    weight
  }
  athletesAggregate {
    aggregate {
      ...AthleteAggregateFieldsFr
    }
    nodes {
      ...AthleteFragment
    }
  }
  displayName
  id
  name
}
}
```

VARIABLES

```
{
  "distinctOn": ["abbreviation"],
  "limit": 987,
  "offset": 123,
  "orderBy": [PositionOrderBy],
  "where": PositionBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```
        "position": [
            {
                "abbreviation": "abc123",
                "athletes": [Athlete],
                "athletesAggregate": Athlet
                "displayName": "xyz789",
                "id": smallint,
                "name": "xyz789"
            }
        ]
    }
```

Subscriptions

predictedPoints

fetch data from the table: "ppa"

Response

Returns [predictedPoints!]!

Arguments

QUERY

```
subscription PredictedPoints(
    $distinctOn: [predictedPointsSele
    $limit: Int,
    $offset: Int,
    $orderBy: [predictedPointsOrderBy
    $where: predictedPointsBoolExp
) {
```

Name	Description
distinctOn - [predictedPointsSelColumn!]	distinct select on [predictedPointsSelColumn!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [predictedPointsOrderBy!]	sort the rows by one or more columns
where - predictedPointsBoolExp	filter the rows

```

predictedPoints(
  distinctOn: $distinctOn,
  limit: $limit,
  offset: $offset,
  orderBy: $orderBy,
  where: $where
) {
  distance
  down
  predictedPoints
  yardLine
}

```

VARIABLES

```
{
  "distinctOn": ["distance"],
  "limit": 123,
  "offset": 987,
  "orderBy": [predictedPointsOrderBy!],
  "where": predictedPointsBoolExp
}
```

RESPONSE

```
{
  "data": {
    "predictedPoints": [
      {
        "distance": smallint,
        "down": smallint,
        "predictedPoints": numeric,
        "yardLine": smallint
      }
    ]
  }
}
```

]}
}
}

Subscriptions

predictedPointsAggregate

fetch aggregated fields from the table:

"ppa"

Response

Returns a [predictedPointsAggregate!](#)

Arguments

Name	Description
distinctOn - [predictedPointsSelection!]	distinct select on
limit - Int	limit the number of rows returned

QUERY

```
subscription PredictedPointsAggrega
  $distinctOn: [predictedPointsSele
  $limit: Int,
  $offset: Int,
  $orderBy: [predictedPointsOrderBy
  $where: predictedPointsBoolExp
) {
  predictedPointsAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
) {
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[predictedPointsOrderBy!]</code>	sort the rows by one or more columns
where - <code>predictedPointsBoolExpr</code>	filter the rows

```

aggregate {
  avg {
    ...predictedPointsAvgFields
  }
  count
  max {
    ...predictedPointsMaxFields
  }
  min {
    ...predictedPointsMinFields
  }
  stddev {
    ...predictedPointsStddevFie
  }
  stddevPop {
    ...predictedPointsStddevPop
  }
  stddevSamp {
    ...predictedPointsStddevSam
  }
  sum {
    ...predictedPointsSumFields
  }
  varPop {
    ...predictedPointsVarPopFie
  }
  varSamp {
    ...predictedPointsVarSampFi
  }
  variance {
    ...predictedPointsVarianceF
  }
}
nodes {
  distance
  down
  predictedPoints
  yardLine
}

```

{
}{
}**VARIABLES**{
 "distinctOn": ["distance"],
 "limit": 123,
 "offset": 123,
 "orderBy": [predictedPointsOrderB
 "where": predictedPointsBoolExp
}
}**RESPONSE**{
 "data": {
 "predictedPointsAggregate": {
 "aggregate": predictedPointsA
 "nodes": [predictedPoints]
 }
 }
}
}

Subscriptions

ratings

fetch data from the table:

"rating_systems"

Response

Returns `[ratings!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[ratingsSelectColumn]</code>	distinct select on columns
<code>limit</code> - <code>Int</code>	limit the number of rows returned
<code>offset</code> - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy</code> - <code>[ratingsOrderBy!]</code>	sort the rows by one or more columns
<code>where</code> - <code>ratingsBoolExp</code>	filter the rows returned

QUERY

```
subscription Ratings(
  $distinctOn: [ratingsSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [ratingsOrderBy!],
  $where: ratingsBoolExp
) {
  ratings(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    conference
    conferenceId
    elo
    fpi
    fpiAvgWinProbabilityRank
    fpiDefensiveEfficiency
    fpiGameControlRank
    fpiOffensiveEfficiency
    fpiOverallEfficiency
    fpiRemainingSosRank
    fpiResumeRank
    fpiSosRank
    fpiSpecialTeamsEfficiency
    fpiStrengthOfRecordRank
  }
}
```

```
        spDefense
        spOffense
        spOverall
        spSpecialTeams
        srs
        team
        teamId
        year
    }
}
```

VARIABLES

```
{
  "distinctOn": ["conference"],
  "limit": 123,
  "offset": 987,
  "orderBy": [ratingsOrderBy],
  "where": ratingsBoolExp
}
```

RESPONSE

```
{
  "data": {
    "ratings": [
      {
        "conference": "abc123",
        "conferenceId": smallint,
        "elo": 123,
        "fpi": numeric,
        "fpiAvgWinProbabilityRank": n
        "fpiDefensiveEfficiency": n
        "fpiGameControlRank": small
        "fpiOffensiveEfficiency": n
        "fpiOverallEfficiency": num
    ]
  }
}
```

```
"fpiRemainingSosRank": smallint,
"fpiResumeRank": smallint,
"fpiSosRank": smallint,
"fpiSpecialTeamsEfficiency": numeric,
"fpiStrengthOfRecordRank": numeric,
"spDefense": numeric,
"spOffense": numeric,
"spOverall": numeric,
"spSpecialTeams": numeric,
"srs": numeric,
"team": "xyz789",
"teamId": 987,
"year": smallint
}
]
}
}
```

Subscriptions

recruit

fetch data from the table: "recruit"

Response

Returns [Recruit!]!

QUERY

```
subscription Recruit(
  $distinctOn: [RecruitSelectColumn]
  $limit: Int,
```

Arguments

Name	Description
distinctOn - [RecruitSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitOrderBy!]	sort the rows by one or more columns
where - RecruitBoolExp	filter the rows returned

```
$offset: Int,
$orderBy: [RecruitOrderBy!],
$where: RecruitBoolExp
) {
  recruit(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    athlete {
      adjustedPlayerMetrics {
        ...AdjustedPlayerMetricsFragment
      }
      adjustedPlayerMetricsAggregate {
        ...AdjustedPlayerMetricsAggregate
      }
      athleteTeams {
        ...AthleteTeamFragment
      }
      athleteTeamsAggregate {
        ...AthleteTeamAggregateFragment
      }
      firstName
      height
      hometown {
        ...HometownFragment
      }
      hometownId
      id
      jersey
      lastName
      name
      position {
        ...PositionFragment
      }
      positionId
      recruits {
        ...RecruitFragment
      }
    }
  }
}
```

```
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
    teamId
    weight
  }
  college {
    abbreviation
    classification
    conference
    conferenceId
    division
    school
    teamId
  }
  height
  hometown {
    athletes {
      ...AthleteFragment
    }
    athletesAggregate {
      ...AthleteAggregateFragment
    }
    city
    country
    countyFips
    latitude
    longitude
    recruits {
      ...RecruitFragment
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
    state
  }
  id
  name
  overallRank
```

```
        position {
          id
          position
          positionGroup
        }
        positionRank
        ranking
        rating
        recruitSchool {
          id
          name
          recruits {
            ...RecruitFragment
          }
          recruitsAggregate {
            ...RecruitAggregateFragment
          }
        }
        recruitType
        stars
        weight
        year
      }
    }
```

VARIABLES

```
{
  "distinctOn": ["height"],
  "limit": 987,
  "offset": 987,
  "orderBy": [RecruitOrderBy],
  "where": RecruitBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "recruit": [  
      {  
        "athlete": Athlete,  
        "college": currentTeams,  
        "height": 987.65,  
        "hometown": Hometown,  
        "id": bigint,  
        "name": "xyz789",  
        "overallRank": smallint,  
        "position": RecruitPosition  
        "positionRank": smallint,  
        "ranking": smallint,  
        "rating": 123.45,  
        "recruitSchool": RecruitSch  
        "recruitType": recruit_type  
        "stars": smallint,  
        "weight": smallint,  
        "year": smallint  
      }  
    ]  
  }  
}
```

Subscriptions

recruitAggregate

fetch aggregated fields from the table:
"recruit"

Response

Returns a [RecruitAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitSelectColumn]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitOrderBy!]	sort the rows by one or more columns
where - RecruitBoolExp	filter the rows returned

QUERY

```
subscription RecruitAggregate(
  $distinctOn: [RecruitSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitOrderBy!],
  $where: RecruitBoolExp
) {
  recruitAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitAvgFieldsFragment
      }
      count
      max {
        ...RecruitMaxFieldsFragment
      }
      min {
        ...RecruitMinFieldsFragment
      }
      stddev {
        ...RecruitStddevFieldsFragm
      }
      stddevPop {
        ...RecruitStddevPopFieldsFr
      }
      stddevSamp {
        ...RecruitStddevSampFieldsF
    
```

```
        }
      sum {
        ...RecruitSumFieldsFragment
      }
      varPop {
        ...RecruitVarPopFieldsFragment
      }
      varSamp {
        ...RecruitVarSampFieldsFragment
      }
      variance {
        ...RecruitVarianceFieldsFragment
      }
    }
  nodes {
    athlete {
      ...AthleteFragment
    }
    college {
      ...currentTeamsFragment
    }
    height
    hometown {
      ...HometownFragment
    }
    id
    name
    overallRank
    position {
      ...RecruitPositionFragment
    }
    positionRank
    ranking
    rating
    recruitSchool {
      ...RecruitSchoolFragment
    }
    recruitType
    stars
    weight
  }
}
```

```
    year
  }
}
}
```

VARIABLES

```
{
  "distinctOn": ["height"],
  "limit": 123,
  "offset": 987,
  "orderBy": [RecruitOrderBy],
  "where": RecruitBoolExp
}
```

RESPONSE

```
{
  "data": {
    "recruitAggregate": {
      "aggregate": RecruitAggregate,
      "nodes": [Recruit]
    }
  }
}
```



Subscriptions

recruitPosition

fetch data from the table:

"recruit_position"

Response

Returns [RecruitPosition!]!

Arguments

Name	Description
distinctOn - [RecruitPositionSelectionColumn!]	distinct select on [RecruitPositionSelectionColumn!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitPositionOrderBy!]	sort the rows by one or more columns
where - RecruitPositionBoolExpr	filter the rows

QUERY

```
subscription RecruitPosition(
  $distinctOn: [RecruitPositionSele
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitPositionOrderBy
  $where: RecruitPositionBoolExp
) {
  recruitPosition(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    position
    positionGroup
  }
}
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 123,
```

```
"offset": 123,  
"orderBy": [RecruitPositionOrderB  
"where": RecruitPositionBoolExp  
}]
```

RESPONSE

```
{  
  "data": {  
    "recruitPosition": [  
      {  
        "id": smallint,  
        "position": "abc123",  
        "positionGroup": "xyz789"  
      }  
    ]  
  }  
}
```



Subscriptions

recruitPositionAggregate

fetch aggregated fields from the table:
"recruit_position"

Response

Returns a [RecruitPositionAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitPositionSelectionColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitPositionOrderBy!]	sort the rows by one or more columns
where - RecruitPositionBoolExpr	filter the rows

QUERY

```

subscription RecruitPositionAggrega
$distinctOn: [RecruitPositionSele
$limit: Int,
$offset: Int,
$orderBy: [RecruitPositionOrderBy
$where: RecruitPositionBoolExp
) {
  recruitPositionAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitPositionAvgFields
      }
      count
      max {
        ...RecruitPositionMaxFields
      }
      min {
        ...RecruitPositionMinFields
      }
      stddev {
        ...RecruitPositionStddevFie
      }
      stdDevPop {
        ...RecruitPositionStddevPop
      }
      stdDevSamp {
        ...RecruitPositionStddevSam
      }
      sum {
        ...RecruitPositionSumFields
      }
    }
  }
}

```

```
    varPop {  
      ...RecruitPositionVarPopFi  
    }  
    varSamp {  
      ...RecruitPositionVarSampFi  
    }  
    variance {  
      ...RecruitPositionVarianceF  
    }  
  }  
  nodes {  
    id  
    position  
    positionGroup  
  }  
}  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [RecruitPositionOrderByB  
  "where": RecruitPositionBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "recruitPositionAggregate": {  
      "aggregate": RecruitPositionA  
      "nodes": [RecruitPosition]
```

{
}
}
}

Subscriptions

recruitSchool

fetch data from the table:

"recruit_school"

Response

Returns [RecruitSchool!]!

Arguments

Name	Description
distinctOn - [RecruitSchoolSelectColumns!]	distinct select on [RecruitSchoolSelectColumns!]
limit - Int	limit the number of rows returned

QUERY

```
subscription RecruitSchool(  
  $distinctOn: [RecruitSchoolSelect  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [RecruitSchoolOrderBy!]!  
  $where: RecruitSchoolBoolExp  
) {  
  recruitSchool(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
    orderBy: $orderBy,  
    where: $where  
) {  
    id
```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[RecruitSchoolOrderBy!]</code>	sort the rows by one or more columns
where - <code>RecruitSchoolBoolExpr!</code>	filter the rows

```

name
recruits {
  athlete {
    ...AthleteFragment
  }
  college {
    ...currentTeamsFragment
  }
  height
  hometown {
    ...HometownFragment
  }
  id
  name
  overallRank
  position {
    ...RecruitPositionFragment
  }
  positionRank
  ranking
  rating
  recruitSchool {
    ...RecruitSchoolFragment
  }
  recruitType
  stars
  weight
  year
}
recruitsAggregate {
  aggregate {
    ...RecruitAggregateFieldsFr
  }
  nodes {
    ...RecruitFragment
  }
}

```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [RecruitSchoolOrderBy]  
  "where": RecruitSchoolBoolExp  
}
```



RESPONSE

```
{  
  "data": {  
    "recruitSchool": [  
      {  
        "id": 123,  
        "name": "xyz789",  
        "recruits": [Recruit],  
        "recruitsAggregate": Recruit  
      }  
    ]  
  }  
}
```



Subscriptions

recruitSchoolAggregate

fetch aggregated fields from the table:
 "recruit_school"

Response

Returns a [RecruitSchoolAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitSchoolSelectColumns!]	distinct select on [RecruitSchoolSelectColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitSchoolOrderBy!]	sort the rows by one or more [RecruitSchoolOrderBy!] columns

QUERY

```

subscription RecruitSchoolAggregate
$distinctOn: [RecruitSchoolSelect
$limit: Int,
$offset: Int,
$orderBy: [RecruitSchoolOrderBy!]
$where: RecruitSchoolBoolExp
) {
  recruitSchoolAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitSchoolAvgFieldsFr
      }
      count
      max {
        ...RecruitSchoolMaxFieldsFr
      }
      min {
        ...RecruitSchoolMinFieldsFr
      }
    }
  }
}
  
```

Name	Description
where -	filter the rows

RecruitSchoolBoolExp returned

```

    stddev {
      ...RecruitSchoolStddevField
    }
    stddevPop {
      ...RecruitSchoolStddevPopField
    }
    stddevSamp {
      ...RecruitSchoolStddevSampField
    }
    sum {
      ...RecruitSchoolSumFieldsFragment
    }
    varPop {
      ...RecruitSchoolVarPopField
    }
    varSamp {
      ...RecruitSchoolVarSampField
    }
    variance {
      ...RecruitSchoolVarianceField
    }
  }
  nodes {
    id
    name
    recruits {
      ...RecruitFragment
    }
    recruitsAggregate {
      ...RecruitAggregateFragment
    }
  }
}

```

VARIABLES

```
{  
  "distinctOn": ["id"],  
  "limit": 987,  
  "offset": 987,  
  "orderBy": [RecruitSchoolOrderBy]  
  "where": RecruitSchoolBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "recruitSchoolAggregate": {  
      "aggregate": RecruitSchoolAgg  
      "nodes": [RecruitSchool]  
    }  
  }  
}
```

Subscriptions

recruitingTeam

fetch data from the table:
"recruiting_team"

Response

Returns [RecruitingTeam!]!

Arguments

Name	Description
distinctOn - [RecruitingTeamSelectColumns!]	distinct select on [RecruitingTeamSelectColumns!]
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitingTeamOrderBy!]	sort the rows by one or more columns
where - RecruitingTeamBoolExp	filter the rows

QUERY

```

subscription RecruitingTeam(
  $distinctOn: [RecruitingTeamSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitingTeamOrderBy!]!
  $where: RecruitingTeamBoolExp
) {
  recruitingTeam(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    id
    points
    rank
    team {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    year
  }
}

```

VARIABLES

```
{
  "distinctOn": ["id"],
```

```
        "limit": 987,  
        "offset": 123,  
        "orderBy": [RecruitingTeamOrderBy  
        "where": RecruitingTeamBoolExp  
    ]}
```

RESPONSE

```
{  
  "data": {  
    "recruitingTeam": [  
      {  
        "id": 987,  
        "points": numeric,  
        "rank": smallint,  
        "team": currentTeams,  
        "year": smallint  
      }  
    ]  
  }  
}
```

Subscriptions

recruitingTeamAggregate

fetch aggregated fields from the table:
"recruiting_team"

Response

Returns a [RecruitingTeamAggregate!](#)

Arguments

Name	Description
distinctOn - [RecruitingTeamSelectColumns!]	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [RecruitingTeamOrderBy!]	sort the rows by one or more columns
where - RecruitingTeamBoolExp	filter the rows

QUERY

```

subscription RecruitingTeamAggregate {
  $distinctOn: [RecruitingTeamSelectColumns!]!
  $limit: Int,
  $offset: Int,
  $orderBy: [RecruitingTeamOrderBy!]!
  $where: RecruitingTeamBoolExp
} {
  recruitingTeamAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...RecruitingTeamAvgFieldsF
      }
      count
      max {
        ...RecruitingTeamMaxFieldsF
      }
      min {
        ...RecruitingTeamMinFieldsF
      }
      stddev {
        ...RecruitingTeamStddevFiel
      }
      stddevPop {
        ...RecruitingTeamStddevPopF
      }
      stddevSamp {
        ...RecruitingTeamStddevSamp
      }
      sum {
        ...RecruitingTeamSumFieldsF
      }
    }
  }
}

```

```
        }
      varPop {
        ...RecruitingTeamVarPopField
      }
      varSamp {
        ...RecruitingTeamVarSampField
      }
      variance {
        ...RecruitingTeamVarianceField
      }
    }
  nodes {
    id
    points
    rank
    team {
      ...currentTeamsFragment
    }
    year
  }
}
```

VARIABLES

```
{
  "distinctOn": ["id"],
  "limit": 987,
  "offset": 987,
  "orderBy": [RecruitingTeamOrderBy],
  "where": RecruitingTeamBoolExp
}
```

RESPONSE

```
{  
  "data": {  
    "recruitingTeamAggregate": {  
      "aggregate": RecruitingTeamAg  
      "nodes": [RecruitingTeam]  
    }  
  }  
}
```

Subscriptions

scoreboard

fetch data from the table: "scoreboard"

Response

Returns `[Scoreboard!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[ScoreboardSelectColumn]s</code>	distinct select on

QUERY

```
subscription Scoreboard(  
  $distinctOn: [ScoreboardSelectCol  
  $limit: Int,  
  $offset: Int,  
  $orderBy: [ScoreboardOrderBy!],  
  $where: ScoreboardBoolExp  
) {  
  scoreboard(  
    distinctOn: $distinctOn,  
    limit: $limit,  
    offset: $offset,  
  )  
}
```

Name	Description
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [ScoreboardOrderBy!]	sort the rows by one or more columns
where - ScoreboardBoolExp	filter the rows returned

```

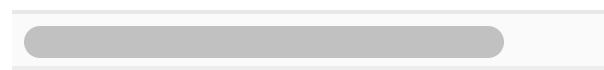
    orderBy: $orderBy,
    where: $where
} {
  awayClassification
  awayConference
  awayConferenceAbbreviation
  awayId
  awayLineScores
  awayPoints
  awayTeam
  city
  conferenceGame
  currentClock
  currentPeriod
  currentPossession
  currentSituation
  homeClassification
  homeConference
  homeConferenceAbbreviation
  homeId
  homeLineScores
  homePoints
  homeTeam
  id
  lastPlay
  moneylineAway
  moneylineHome
  neutralSite
  overUnder
  spread
  startDate
  startTimeTbd
  state
  status
  temperature
  tv
  venue
  weatherDescription
  windDirection
  windSpeed
}

```

```
    }  
}
```

VARIABLES

```
{  
  "distinctOn": ["awayClassification"],  
  "limit": 123,  
  "offset": 123,  
  "orderBy": [ScoreboardOrderBy],  
  "where": ScoreboardBoolExp  
}
```



RESPONSE

```
{  
  "data": {  
    "scoreboard": [  
      {  
        "awayClassification": divis,  
        "awayConference": "xyz789",  
        "awayConferenceAbbreviation": "xyz789",  
        "awayId": 987,  
        "awayLineScores": [smallint],  
        "awayPoints": smallint,  
        "awayTeam": "abc123",  
        "city": "xyz789",  
        "conferenceGame": true,  
        "currentClock": "xyz789",  
        "currentPeriod": smallint,  
        "currentPossession": "abc123",  
        "currentSituation": "abc123",  
        "homeClassification": divis,  
        "homeConference": "abc123",  
        "homeConferenceAbbreviation": "xyz789",  
        "homeId": 987,  
        "homeLineScores": [smallint],  
        "homePoints": smallint,  
        "homeTeam": "abc123",  
        "isHome": true,  
        "period": 1,  
        "possession": "abc123",  
        "situation": "abc123",  
        "team": "abc123",  
        "time": "xyz789"  
      }  
    ]  
  }  
}
```

```
        "homeId": 123,  
        "homeLineScores": [smallint  
        "homePoints": smallint,  
        "homeTeam": "xyz789",  
        "id": 987,  
        "lastPlay": "abc123",  
        "moneylineAway": 987,  
        "moneylineHome": 987,  
        "neutralSite": false,  
        "overUnder": numeric,  
        "spread": numeric,  
        "startDate": timestamptz,  
        "startTimeTbd": false,  
        "state": "abc123",  
        "status": game_status,  
        "temperature": numeric,  
        "tv": "abc123",  
        "venue": "xyz789",  
        "weatherDescription": "xyz7  
        "windDirection": numeric,  
        "windSpeed": numeric  
    }  
]  
}  
}
```

Subscriptions

teamTalent

fetch data from the table: "team_talent"

Response

Returns `[TeamTalent!]!`

Arguments

Name	Description
<code>distinctOn</code> - <code>[TeamTalentSelectColumn!]</code>	distinct select on
<code>limit</code> - <code>Int</code>	limit the number of rows returned
<code>offset</code> - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy</code> - <code>[TeamTalentOrderBy!]</code>	sort the rows by one or more columns
<code>where</code> - <code>TeamTalentBoolExp</code>	filter the rows returned

QUERY

```
subscription TeamTalent(
  $distinctOn: [TeamTalentSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [TeamTalentOrderBy!],
  $where: TeamTalentBoolExp
) {
  teamTalent(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    talent
    team {
      abbreviation
      classification
      conference
      conferenceId
      division
      school
      teamId
    }
    year
  }
}
```

VARIABLES

```
{  
  "distinctOn": ["talent"],  
  "limit": 987,  
  "offset": 123,  
  "orderBy": [TeamTalentOrderBy],  
  "where": TeamTalentBoolExp  
}
```

RESPONSE

```
{  
  "data": {  
    "teamTalent": [  
      {  
        "talent": numeric,  
        "team": currentTeams,  
        "year": smallint  
      }  
    ]  
  }  
}
```

Subscriptions

teamTalentAggregate

fetch aggregated fields from the table:
"team_talent"

Response

Returns a [TeamTalentAggregate!](#)

Arguments

Name	Description
distinctOn - [TeamTalentSelectColumn!] s	distinct select on
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by
orderBy - [TeamTalentOrderBy!]	sort the rows by one or more columns
where - TeamTalentBoolExp	filter the rows returned

QUERY

```
subscription TeamTalentAggregate(
  $distinctOn: [TeamTalentSelectCol
  $limit: Int,
  $offset: Int,
  $orderBy: [TeamTalentOrderBy!],
  $where: TeamTalentBoolExp
) {
  teamTalentAggregate(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    aggregate {
      avg {
        ...TeamTalentAvgFieldsFragm
      }
      count
      max {
        ...TeamTalentMaxFieldsFragm
      }
      min {
        ...TeamTalentMinFieldsFragm
      }
      stdDev {
        ...TeamTalentStddevFieldsFr
      }
      stdDevPop {
        ...TeamTalentStddevPopField
      }
      stdDevSamp {
        ...TeamTalentStddevSampFiel
      }
      sum {
        ...TeamTalentSumFieldsFragm
      }
    }
  }
}
```

```
        }
      varPop {
        ...TeamTalentVarPopFieldsFr
      }
      varSamp {
        ...TeamTalentVarSampFieldsF
      }
      variance {
        ...TeamTalentVarianceFields
      }
    }
    nodes {
      talent
      team {
        ...currentTeamsFragment
      }
      year
    }
  }
}
```

VARIABLES

```
{
  "distinctOn": ["talent"],
  "limit": 987,
  "offset": 123,
  "orderBy": [TeamTalentOrderBy],
  "where": TeamTalentBoolExp
}
```

RESPONSE

```
{
  "data": {
```

```

    "teamTalentAggregate": {
      "aggregate": TeamTalentAggreg
      "nodes": [TeamTalent]
    }
  }
}

```

Subscriptions

transfer

fetch data from the table: "transfer"

Response

Returns `[Transfer!]!`

Arguments

Name	Description
<code>distinctOn - [TransferSelectColumn]</code>	distinct select on <code>[TransferSelectColumn]</code>
<code>limit - Int</code>	limit the number of rows returned

QUERY

```

subscription Transfer(
  $distinctOn: [TransferSelectColumn]
  $limit: Int,
  $offset: Int,
  $orderBy: [TransferOrderBy!],
  $where: TransferBoolExp
) {
  transfer(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  )
}

```

Name	Description
offset - <code>Int</code>	skip the first n rows. Use only with <code>order_by</code>
orderBy - <code>[TransferOrderBy!]</code>	sort the rows by one or more columns
where - <code>TransferBoolExp</code>	filter the rows returned

```

    } {
      eligibility
      firstName
      fromTeam {
        abbreviation
        classification
        conference
        conferenceId
        division
        school
        teamId
      }
      lastName
      position {
        id
        position
        positionGroup
      }
      rating
      season
      stars
      toTeam {
        abbreviation
        classification
        conference
        conferenceId
        division
        school
        teamId
      }
      transferDate
    }
  }
}

```

VARIABLES

```
{
  "distinctOn": ["eligibility"],
```

```
        "limit": 123,  
        "offset": 987,  
        "orderBy": [TransferOrderBy],  
        "where": TransferBoolExp  
    }  
}
```

RESPONSE

```
{  
  "data": {  
    "transfer": [  
      {  
        "eligibility": "xyz789",  
        "firstName": "xyz789",  
        "fromTeam": currentTeams,  
        "lastName": "xyz789",  
        "position": RecruitPosition,  
        "rating": numeric,  
        "season": smallint,  
        "stars": smallint,  
        "toTeam": currentTeams,  
        "transferDate": timestamp  
      }  
    ]  
  }  
}
```



Subscriptions

weatherCondition

fetch data from the table:

"weather_condition"

Response

Returns `[WeatherCondition!]!`

Arguments

Name	Description
<code>distinctOn - [WeatherConditionSelectionColumn!]</code>	distinct select on
<code>limit - Int</code>	limit the number of rows returned
<code>offset - Int</code>	skip the first n rows. Use only with <code>order_by</code>
<code>orderBy - [WeatherConditionOrderBy!]</code>	sort the rows by one or more columns
<code>where - WeatherConditionBoolExp</code>	filter the rows

QUERY

```
subscription WeatherCondition(
  $distinctOn: [WeatherConditionSel
  $limit: Int,
  $offset: Int,
  $orderBy: [WeatherConditionOrderB
  $where: WeatherConditionBoolExp
) {
  weatherCondition(
    distinctOn: $distinctOn,
    limit: $limit,
    offset: $offset,
    orderBy: $orderBy,
    where: $where
  ) {
    description
    id
  }
}
```

VARIABLES

```
{
  "distinctOn": ["description"],
  "limit": 123,
  "offset": 123,
```

```
"orderBy": [WeatherConditionOrder  
"where": WeatherConditionBoolExp  
}]
```

RESPONSE

```
{  
  "data": {  
    "weatherCondition": [  
      {  
        "description": "xyz789",  
        "id": smallint  
      }  
    ]  
  }  
}
```

Types

AdjustedPlayerMetrics

columns and relationships of
"adjusted_player_metrics"

EXAMPLE

```
{  
  "athlete": Athlete,  
  "athleteId": bigint,  
  "metricType": player_adjusted_met  
  "metricValue": numeric,
```

Field Name	Description	
athlete - Athlete!	An object relationship	"plays": smallint, "year": smallint }
athleteId - bigint!		
metricType - player_adjusted_metric_type!		
metricValue - numeric!		
plays - smallint		
year - smallint!		

Types

AdjustedPlayerMetricsAggregate

aggregated selection of "adjusted_player_metrics"

EXAMPLE

```
{
  "aggregate": AdjustedPlayerMetric
```

Field Name	Description	
aggregate -		"nodes": [AdjustedPlayerMetrics]
nodes -	AdjustedPlayerMetricsAggregateFields	}

Types

AdjustedPlayerMetricsAggregateBo...

Input Field	Description	EXAMPLE
count -		{"count": adjustedPlayerMetricsAggr}

Types

AdjustedPlayerMetricsAggregateFie...

aggregate fields of
"adjusted_player_metrics"

Field Name	Description
avg -	AdjustedPlayerMetricsAvgFields
count - Int!	
	Arguments
columns -	[AdjustedPlayerMetricsSelectColumn!]
distinct - Boolean	
max -	AdjustedPlayerMetricsMaxFields
min -	AdjustedPlayerMetricsMinFields
stddev -	AdjustedPlayerMetricsStddevFields
stddevPop -	AdjustedPlayerMetricsStddevPopFields
stddevSamp -	AdjustedPlayerMetricsStddevSampFields

EXAMPLE

```
{  
  "avg": AdjustedPlayerMetricsAvgFi  
  "count": 987,  
  "max": AdjustedPlayerMetricsMaxFi  
  "min": AdjustedPlayerMetricsMinFi  
  "stddev": AdjustedPlayerMetricsSt  
  "stddevPop": AdjustedPlayerMetric  
  "stddevSamp": AdjustedPlayerMetri  
  "sum": AdjustedPlayerMetricsSumFi  
  "varPop": AdjustedPlayerMetricsVa  
  "varSamp": AdjustedPlayerMetricsV  
  "variance": AdjustedPlayerMetrics
```

}

Field Name	Description
sum -	AdjustedPlayerMetricsSumFields
varPop -	AdjustedPlayerMetricsVarPopFields
varSamp -	AdjustedPlayerMetricsVarSampFields
variance -	AdjustedPlayerMetricsVarianceFields

Types

AdjustedPlayerMetricsAggregateOr...

order by aggregate values of table
"adjusted_player_metrics"

EXAMPLE

```
{
  "avg": AdjustedPlayerMetricsAvgOr
  "count": "ASC",
  "max": AdjustedPlayerMetricsMaxOr
  "min": AdjustedPlayerMetricsMinOr
  "stddev": AdjustedPlayerMetricsSt
  "stddevPop": AdjustedPlayerMetric
  "stddevSamp": AdjustedPlayerMetri
  "sum": AdjustedPlayerMetricsSumOr
```

Input Field	Description
avg -	AdjustedPlayerMetricsAvgOrderBy
count - OrderBy	

Input Field	Description	
max -	AdjustedPlayerMetricsMaxOrderBy	"varPop": AdjustedPlayerMetricsVa "varSamp": AdjustedPlayerMetricsV "variance": AdjustedPlayerMetrics
min -	AdjustedPlayerMetricsMinOrderBy	}
stddev -	AdjustedPlayerMetricsStddevOrderBy	
stddevPop -	AdjustedPlayerMetricsStddevPopOrderBy	
stddevSamp -	AdjustedPlayerMetricsStddevSampOrderBy	
sum -	AdjustedPlayerMetricsSumOrderBy	
varPop -	AdjustedPlayerMetricsVarPopOrderBy	
varSamp -	AdjustedPlayerMetricsVarSampOrderBy	
variance -	AdjustedPlayerMetricsVarianceOrderBy	

Types

AdjustedPlayerMetricsAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
athleteId - Float	
metricValue - Float	
plays - Float	
year - Float	

```
{"athleteId": 123.45, "metricValue":
```

Types

AdjustedPlayerMetricsAvgOrderBy

order by avg() on columns of table
"adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays -	OrderBy
year -	OrderBy

Types

AdjustedPlayerMetricsBoolExp

Boolean expression to filter rows from the table "adjusted_player_metrics". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	
[AdjustedPlayerMetricsBoolExp!]	
_not -	
AdjustedPlayerMetricsBoolExp	

EXAMPLE

```
{
  "_and": [AdjustedPlayerMetricsBoolExp!],
  "_not": AdjustedPlayerMetricsBoolExp,
  "_or": [AdjustedPlayerMetricsBoolExp!],
  "athlete": AthleteBoolExp,
  "athleteId": BigintComparisonExp,
  "metricType": PlayerAdjustedMetricType,
  "metricValue": NumericComparisonExp,
  "plays": SmallintComparisonExp,
```

Input Field	Description	
_or -		"year": SmallintComparisonExp
	[AdjustedPlayerMetricsBoolExp!]	}
athlete -		
	AthleteBoolExp	
athleteId -		
	BigintComparisonExp	
metricType -		
	PlayerAdjustedMetricTypeComparisonExp	
metricValue -		
	NumericComparisonExp	
plays -		
	SmallintComparisonExp	
year -		
	SmallintComparisonExp	

Types

AdjustedPlayerMetricsMaxFields

aggregate max on columns

Field Name	Description
athleteId - bigint	
metricType -	
	player_adjusted_metric_type
metricValue -	
	numeric
plays - smallint	
year - smallint	

EXAMPLE

```
{
  "athleteId": bigint,
  "metricType": player_adjusted_metric_type,
  "metricValue": numeric,
  "plays": smallint,
  "year": smallint
}
```



Types

AdjustedPlayerMetricsMaxOrderBy

order by max() on columns of table
 "adjusted_player_metrics"

EXAMPLE

```
{
  "athleteId": "ASC",
  "metricType": "ASC",
  "metricValue": "ASC",
  "plays": "ASC",
```

Input Field	Description	
athleteId -		"year": "ASC"
OrderBy		}
metricType -		
OrderBy		
metricValue -		
OrderBy		
plays - OrderBy		
year - OrderBy		

Types

AdjustedPlayerMetricsMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
athleteId - bigint	
metricType -	
player_adjusted_metric_type	

```
{
  "athleteId": bigint,
  "metricType": player_adjusted_met
  "metricValue": numeric,
  "plays": smallint,
```

Field Name	Description	
metricValue -		"year": smallint
numeric		}
plays - smallint		
year - smallint		

Types

AdjustedPlayerMetricsMinOrderBy

order by min() on columns of table
 "adjusted_player_metrics"

EXAMPLE

```
{
  "athleteId": "ASC",
  "metricType": "ASC",
  "metricValue": "ASC",
  "plays": "ASC",
  "year": "ASC"
}
```

Input Field	Description
athleteId -	
OrderBy	
metricType -	
OrderBy	
metricValue -	
OrderBy	

Input Field	Description
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsOrderBy

Ordering options when selecting data from "adjusted_player_metrics".

EXAMPLE

```
{  
  "athlete": AthleteOrderBy,  
  "athleteId": "ASC",  
  "metricType": "ASC",  
  "metricValue": "ASC",  
  "plays": "ASC",  
  "year": "ASC"  
}
```

Input Field	Description
athlete - AthleteOrderBy	
athleteId - OrderBy	
metricType - OrderBy	
metricValue - OrderBy	

Input Field	Description
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsSelectColumn

select columns of table

EXAMPLE

"adjusted_player_metrics"

["athleteId"](#)

Enum Value	Description
athleteId	column name
metricType	column name
metricValue	column name
plays	column name
year	column name

Types

AdjustedPlayerMetricsStddevFields

aggregate stddev on columns

EXAMPLE

```
{"athleteId": 123.45, "metricValue":
```

Field Name	Description
athleteId - Float	
metricValue - Float	
plays - Float	
year - Float	

Types

AdjustedPlayerMetricsStddevOrder...

order by stddev() on columns of table

EXAMPLE

```
"adjusted_player_metrics"
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

```
{"athleteId": "ASC", "metricValue":
```

Types

AdjustedPlayerMetricsStddevPopFi...

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
athleteId - Float	
metricValue -	
Float	
plays - Float	
year - Float	

```
{"athleteId": 123.45, "metricValue":
```

Types

AdjustedPlayerMetricsStddevPopOr...

order by stddevPop() on columns of
table "adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsStddevSamp...

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
athleteId - Float	
metricValue - Float	
plays - Float	
year - Float	

```
{"athleteId": 123.45, "metricValue":
```

Types

AdjustedPlayerMetricsStddevSamp...

order by stddevSamp() on columns of
table "adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
athleteId - bigrnt	
metricValue -	
numeric	
plays - smallint	
year - smallint	

```
{  
    "athleteId": bigrnt,  
    "metricValue": numeric,  
    "plays": smallint,  
    "year": smallint  
}
```

Types

AdjustedPlayerMetricsSumOrderBy

order by sum() on columns of table
"adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
athleteId - Float	
metricValue - Float	
plays - Float	
year - Float	

```
{"athleteId": 123.45, "metricValue":
```

Types

AdjustedPlayerMetricsVarPopOrder...

order by varPop() on columns of table
"adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsVarSampFields

aggregate varSamp on columns

EXAMPLE

```
{"athleteId": 123.45, "metricValue":
```

Field Name	Description
athleteId - Float	
metricValue -	
Float	
plays - Float	
year - Float	

Types

AdjustedPlayerMetricsVarSampOrd...

order by varSamp() on columns of table
"adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedPlayerMetricsVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
athleteId - Float	
metricValue - Float	
plays - Float	
year - Float	

```
{"athleteId": 987.65, "metricValue":
```

Types

AdjustedPlayerMetricsVarianceOrd...

order by variance() on columns of table
"adjusted_player_metrics"

EXAMPLE

```
{"athleteId": "ASC", "metricValue":
```

Input Field	Description
athleteId -	
OrderBy	
metricValue -	
OrderBy	
plays - OrderBy	
year - OrderBy	

Types

AdjustedTeamMetrics

columns and relationships of
 "adjusted_team_metrics"

Field Name	Description
epa - numeric!	
epaAllowed -	
numeric!	

EXAMPLE

```
{
  "epa": numeric,
  "epaAllowed": numeric,
  "explosiveness": numeric,
  "explosivenessAllowed": numeric,
  "highlightYards": numeric,
  "highlightYardsAllowed": numeric,
  "lineYards": numeric,
  "lineYardsAllowed": numeric,
  "openFieldYards": numeric,
  "openFieldYardsAllowed": numeric,
  "passingDownsSuccess": numeric,
```

Field Name	Description
explosiveness - <code>numeric!</code>	
explosivenessAllowed - <code>numeric!</code>	
highlightYards - <code>numeric!</code>	
highlightYardsAllowed - <code>numeric!</code>	
lineYards - <code>numeric!</code>	
lineYardsAllowed - <code>numeric!</code>	
openFieldYards - <code>numeric!</code>	
openFieldYardsAllowed - <code>numeric!</code>	
passingDownsSuccess - <code>numeric!</code>	
passingDownsSuccessAllowed - <code>numeric!</code>	
passingEpa - <code>numeric!</code>	
	<code>"passingDownsSuccessAllowed": num</code> <code>"passingEpa": numeric,</code> <code>"passingEpaAllowed": numeric,</code> <code>"rushingEpa": numeric,</code> <code>"rushingEpaAllowed": numeric,</code> <code>"secondLevelYards": numeric,</code> <code>"secondLevelYardsAllowed": numeric</code> <code>"standardDownsSuccess": numeric,</code> <code>"standardDownsSuccessAllowed": numeric</code> <code>"success": numeric,</code> <code>"successAllowed": numeric,</code> <code>"team": currentTeams,</code> <code>"teamId": 123,</code> <code>"year": smallint</code>

Field Name	Description
passingEpaAllowed	- <code>numeric!</code>
rushingEpa	- <code>numeric!</code>
rushingEpaAllowed	- <code>numeric!</code>
secondLevelYards	- <code>numeric!</code>
secondLevelYardsAllowed	- <code>numeric!</code>
standardDownsSuccess	- <code>numeric!</code>
standardDownsSuccessAllowed	- <code>numeric!</code>
success	- <code>numeric!</code>
successAllowed	- <code>numeric!</code>
team	An object
<code>currentTeams</code>	relationship
teamId	- <code>Int!</code>
year	- <code>smallint!</code>

Types

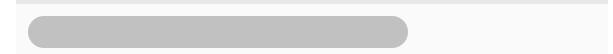
AdjustedTeamMetricsAggregate

aggregated selection of
"adjusted_team_metrics"

Field Name	Description
aggregate -	AdjustedTeamMetricsAggregateFields
nodes -	[AdjustedTeamMetrics!]!

EXAMPLE

```
{  
  "aggregate": AdjustedTeamMetricsA  
  "nodes": [AdjustedTeamMetrics]  
}
```



Types

AdjustedTeamMetricsAggregateFiel...

aggregate fields of
"adjusted_team_metrics"

EXAMPLE

```
{  
  "avg": AdjustedTeamMetricsAvgFiel  
  "count": 123,  
  "min": ...  
}
```

Field Name	Description
avg -	
	AdjustedTeamMetricsAvgFields
count - Int!	
	<div style="border: 1px solid #ccc; padding: 5px;"> Arguments </div>
columns -	
	[AdjustedTeamMetricsSelectColumn!]
distinct - Boolean	
max -	
	AdjustedTeamMetricsMaxFields
min -	
	AdjustedTeamMetricsMinFields
stddev -	
	AdjustedTeamMetricsStddevFields
stddevPop -	
	AdjustedTeamMetricsStddevPopFields
stddevSamp -	
	AdjustedTeamMetricsStddevSampFields
sum -	
	AdjustedTeamMetricsSumFields
varPop -	
	AdjustedTeamMetricsVarPopFields

```

    "max": AdjustedTeamMetricsMaxFiel
    "min": AdjustedTeamMetricsMinFiel
    "stddev": AdjustedTeamMetricsStdd
    "stddevPop": AdjustedTeamMetricsS
    "stddevSamp": AdjustedTeamMetrics
    "sum": AdjustedTeamMetricsSumFiel
    "varPop": AdjustedTeamMetricsVarP
    "varSamp": AdjustedTeamMetricsVar
    "variance": AdjustedTeamMetricsVa
}
  
```

Field Name	Description
varSamp -	AdjustedTeamMetricsVarSampFields
variance -	AdjustedTeamMetricsVarianceFields

Types

AdjustedTeamMetricsAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
epa - Float	
epaAllowed - Float	
explosiveness -	
Float	
explosivenessAllowed	
- Float	

```
{
  "epa": 123.45,
  "epaAllowed": 123.45,
  "explosiveness": 987.65,
  "explosivenessAllowed": 987.65,
  "highlightYards": 987.65,
  "highlightYardsAllowed": 987.65,
  "lineYards": 987.65,
  "lineYardsAllowed": 123.45,
  "openFieldYards": 123.45,
  "openFieldYardsAllowed": 987.65,
  "passingDownsSuccess": 987.65,
  "passingDownsSuccessAllowed": 123
  "passingEpa": 987.65,
  "passingEpaAllowed": 987.65,
  "rushingEpa": 987.65,
```

Field Name	Description
highlightYards -	
Float	
highlightYardsAllowed	
- Float	
lineYards -	
Float	
lineYardsAllowed	
- Float	
openFieldYards -	
Float	
openFieldYardsAllowed	
- Float	
passingDownsSuccess	
- Float	
passingDownsSuccessAllowed	
- Float	
passingEpa -	
Float	
passingEpaAllowed	
- Float	
rushingEpa -	
Float	
rushingEpaAllowed	
- Float	

```
"rushingEpaAllowed": 123.45,  
"secondLevelYards": 123.45,  
"secondLevelYardsAllowed": 987.65  
"standardDownsSuccess": 987.65,  
"standardDownsSuccessAllowed": 12  
"success": 987.65,  
"successAllowed": 987.65,  
"teamId": 987.65,  
"year": 123.45  
}
```

Field Name	Description
secondLevelYards	- Float
secondLevelYardsAllowed	- Float
standardDownsSuccess	- Float
standardDownsSuccessAllowed	- Float
success	- Float
successAllowed	- Float
teamId	- Float
year	- Float

Types

AdjustedTeamMetricsBoolExp

Boolean expression to filter rows from the table "adjusted_team_metrics". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	<code>[AdjustedTeamMetricsBoolExp!]</code>
_not -	<code>AdjustedTeamMetricsBoolExp</code>
_or -	<code>[AdjustedTeamMetricsBoolExp!]</code>
epa -	<code>NumericComparisonExp</code>
epaAllowed -	<code>NumericComparisonExp</code>
explosiveness -	<code>NumericComparisonExp</code>
explosivenessAllowed -	<code>NumericComparisonExp</code>
highlightYards -	<code>NumericComparisonExp</code>
highlightYardsAllowed -	

EXAMPLE

```
{
  "_and": [AdjustedTeamMetricsBoolExp],
  "_not": AdjustedTeamMetricsBoolExp,
  "_or": [AdjustedTeamMetricsBoolExp],
  "epa": NumericComparisonExp,
  "epaAllowed": NumericComparisonExp,
  "explosiveness": NumericComparisonExp,
  "explosivenessAllowed": NumericComparisonExp,
  "highlightYards": NumericComparisonExp,
  "highlightYardsAllowed": NumericComparisonExp,
  "lineYards": NumericComparisonExp,
  "lineYardsAllowed": NumericComparisonExp,
  "openFieldYards": NumericComparisonExp,
  "openFieldYardsAllowed": NumericComparisonExp,
  "passingDownsSuccess": NumericComparisonExp,
  "passingDownsSuccessAllowed": NumericComparisonExp,
  "passingEpa": NumericComparisonExp,
  "passingEpaAllowed": NumericComparisonExp,
  "rushingEpa": NumericComparisonExp,
  "rushingEpaAllowed": NumericComparisonExp,
  "secondLevelYards": NumericComparisonExp,
  "secondLevelYardsAllowed": NumericComparisonExp,
  "standardDownsSuccess": NumericComparisonExp,
  "standardDownsSuccessAllowed": NumericComparisonExp,
  "success": NumericComparisonExp,
  "successAllowed": NumericComparisonExp,
  "team": currentTeamsBoolExp,
  "teamId": IntComparisonExp,
  "year": SmallIntComparisonExp
}
```

Input Field	Description
NumericComparisonExp	
lineYards -	
NumericComparisonExp	
lineYardsAllowed	
-	
NumericComparisonExp	
openFieldYards -	
NumericComparisonExp	
openFieldYardsAllowed	
-	
NumericComparisonExp	
passingDownsSuccess	
-	
NumericComparisonExp	
passingDownsSuccessAllowed	
-	
NumericComparisonExp	
passingEpa -	
NumericComparisonExp	
passingEpaAllowed	
-	
NumericComparisonExp	

Input Field	Description
rushingEpa -	
	NumericComparisonExp
rushingEpaAllowed	-
	NumericComparisonExp
secondLevelYards	-
	NumericComparisonExp
secondLevelYardsAllowed	-
	NumericComparisonExp
standardDownsSuccess	-
	NumericComparisonExp
standardDownsSuccessAllowed	-
	NumericComparisonExp
success -	
	NumericComparisonExp
successAllowed -	
	NumericComparisonExp
team -	
	currentTeamsBoolExp

Input Field	Description
teamId -	
	IntComparisonExp
year -	
	SmallIntComparisonExp

Types

AdjustedTeamMetricsMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
epa - numeric	
epaAllowed -	
	numeric
explosiveness -	
	numeric
explosivenessAllowed	
- numeric	

```
{
  "epa": numeric,
  "epaAllowed": numeric,
  "explosiveness": numeric,
  "explosivenessAllowed": numeric,
  "highlightYards": numeric,
  "highlightYardsAllowed": numeric,
  "lineYards": numeric,
  "lineYardsAllowed": numeric,
  "openFieldYards": numeric,
  "openFieldYardsAllowed": numeric,
  "passingDownsSuccess": numeric,
  "passingDownsSuccessAllowed": num
  "passingEpa": numeric,
  "passingEpaAllowed": numeric,
  "rushingEpa": numeric,
```

Field Name	Description
highlightYards - numeric	
highlightYardsAllowed - numeric	
lineYards - numeric	
lineYardsAllowed - numeric	
openFieldYards - numeric	
openFieldYardsAllowed - numeric	
passingDownsSuccess - numeric	
passingDownsSuccessAllowed - numeric	
passingEpa - numeric	
passingEpaAllowed - numeric	
rushingEpa - numeric	

```
"rushingEpaAllowed": numeric,  
"secondLevelYards": numeric,  
"secondLevelYardsAllowed": numeric  
"standardDownsSuccess": numeric,  
"standardDownsSuccessAllowed": numeric  
"success": numeric,  
"successAllowed": numeric,  
"teamId": 123,  
"year": smallint  
}
```

Field Name	Description
rushingEpaAllowed	- numeric
secondLevelYards	- numeric
secondLevelYardsAllowed	- numeric
standardDownsSuccess	- numeric
standardDownsSuccessAllowed	- numeric
success - numeric	
successAllowed - numeric	
teamId - Int	
year - smallint	

Types

AdjustedTeamMetricsMinFields

aggregate min on columns

Field Name	Description
epa - numeric	
epaAllowed - numeric	
explosiveness - numeric	
explosivenessAllowed - numeric	
highlightYards - numeric	
highlightYardsAllowed - numeric	
lineYards - numeric	
lineYardsAllowed - numeric	
openFieldYards - numeric	
openFieldYardsAllowed - numeric	

EXAMPLE

```
{
  "epa": numeric,
  "epaAllowed": numeric,
  "explosiveness": numeric,
  "explosivenessAllowed": numeric,
  "highlightYards": numeric,
  "highlightYardsAllowed": numeric,
  "lineYards": numeric,
  "lineYardsAllowed": numeric,
  "openFieldYards": numeric,
  "openFieldYardsAllowed": numeric,
  "passingDownsSuccess": numeric,
  "passingDownsSuccessAllowed": num
  "passingEpa": numeric,
  "passingEpaAllowed": numeric,
  "rushingEpa": numeric,
  "rushingEpaAllowed": numeric,
  "secondLevelYards": numeric,
  "secondLevelYardsAllowed": numeri
  "standardDownsSuccess": numeric,
  "standardDownsSuccessAllowed": nu
  "success": numeric,
  "successAllowed": numeric,
  "teamId": 987,
  "year": smallint
}
```



Field Name	Description
passingDownsSuccess	- numeric
passingDownsSuccessAllowed	- numeric
passingEpa	- numeric
passingEpaAllowed	- numeric
rushingEpa	- numeric
rushingEpaAllowed	- numeric
secondLevelYards	- numeric
secondLevelYardsAllowed	- numeric
standardDownsSuccess	- numeric
standardDownsSuccessAllowed	- numeric
success	- numeric

Field Name	Description
successAllowed -	
numeric	
teamId -	Int
year -	smallint

Types

AdjustedTeamMetricsOrderBy

Ordering options when selecting data from "adjusted_team_metrics".

Input Field	Description
epa -	OrderBy
epaAllowed -	OrderBy
explosiveness -	OrderBy

EXAMPLE

```
{
  "epa": "ASC",
  "epaAllowed": "ASC",
  "explosiveness": "ASC",
  "explosivenessAllowed": "ASC",
  "highlightYards": "ASC",
  "highlightYardsAllowed": "ASC",
  "lineYards": "ASC",
  "lineYardsAllowed": "ASC",
  "openFieldYards": "ASC",
  "openFieldYardsAllowed": "ASC",
  "passingDownsSuccess": "ASC",
  "passingDownsSuccessAllowed": "AS
  "passingEpa": "ASC",
  "passingEpaAllowed": "ASC",
```

Input Field	Description
explosivenessAllowed	
- OrderBy	
highlightYards -	
OrderBy	
highlightYardsAllowed	
- OrderBy	
lineYards -	
OrderBy	
lineYardsAllowed	
- OrderBy	
openFieldYards -	
OrderBy	
openFieldYardsAllowed	
- OrderBy	
passingDownsSuccess	
- OrderBy	
passingDownsSuccessAllowed	
- OrderBy	
passingEpa -	
OrderBy	
passingEpaAllowed	
- OrderBy	

```

"rushingEpa": "ASC",
"rushingEpaAllowed": "ASC",
"secondLevelYards": "ASC",
"secondLevelYardsAllowed": "ASC",
"standardDownsSuccess": "ASC",
"standardDownsSuccessAllowed": "A
"success": "ASC",
"successAllowed": "ASC",
"team": currentTeamsOrderBy,
"teamId": "ASC",
"year": "ASC"
}

```

Input Field	Description
rushingEpa -	
OrderBy	
rushingEpaAllowed	
- OrderBy	
secondLevelYards	
- OrderBy	
secondLevelYardsAllowed	
- OrderBy	
standardDownsSuccess	
- OrderBy	
standardDownsSuccessAllowed	
- OrderBy	
success - OrderBy	
successAllowed -	
OrderBy	
team -	
currentTeamsOrderBy	
teamId - OrderBy	
year - OrderBy	

Types

AdjustedTeamMetricsSelectColumn

select columns of table

"adjusted_team_metrics"

EXAMPLE

"epa"

Enum Value	Description
epa	column name
epaAllowed	column name
explosiveness	column name
explosivenessAllowed	column name
highlightYards	column name
highlightYardsAllowed	column name
lineYards	column name
lineYardsAllowed	column name
openFieldYards	column name
openFieldYardsAllowed	column name
passingDownsSuccess	column name
passingDownsSuccessAllowed	column name
passingEpa	column name

Enum Value	Description
passingEpaAllowed	column name
rushingEpa	column name
rushingEpaAllowed	column name
secondLevelYards	column name
secondLevelYardsAllowed	column name
standardDownsSuccess	column name
standardDownsSuccessAllowed	column name
success	column name
successAllowed	column name
teamId	column name
year	column name

Types

AdjustedTeamMetricsStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
epa - Float	
epaAllowed - Float	
explosiveness - Float	
explosivenessAllowed - Float	
highlightYards - Float	
highlightYardsAllowed - Float	
lineYards - Float	
lineYardsAllowed - Float	
openFieldYards - Float	
openFieldYardsAllowed - Float	
passingDownsSuccess - Float	
passingDownsSuccessAllowed - Float	

```
{  
  "epa": 123.45,  
  "epaAllowed": 123.45,  
  "explosiveness": 987.65,  
  "explosivenessAllowed": 123.45,  
  "highlightYards": 987.65,  
  "highlightYardsAllowed": 987.65,  
  "lineYards": 987.65,  
  "lineYardsAllowed": 987.65,  
  "openFieldYards": 987.65,  
  "openFieldYardsAllowed": 987.65,  
  "passingDownsSuccess": 123.45,  
  "passingDownsSuccessAllowed": 123.45,  
  "passingEpa": 123.45,  
  "passingEpaAllowed": 987.65,  
  "rushingEpa": 123.45,  
  "rushingEpaAllowed": 987.65,  
  "secondLevelYards": 987.65,  
  "secondLevelYardsAllowed": 987.65,  
  "standardDownsSuccess": 123.45,  
  "standardDownsSuccessAllowed": 123.45,  
  "success": 123.45,  
  "successAllowed": 987.65,  
  "teamId": 987.65,  
  "year": 123.45  
}
```

Field Name	Description
passingEpa - Float	
passingEpaAllowed - Float	
rushingEpa - Float	
rushingEpaAllowed - Float	
secondLevelYards - Float	
secondLevelYardsAllowed - Float	
standardDownsSuccess - Float	
standardDownsSuccessAllowed - Float	
success - Float	
successAllowed - Float	
teamId - Float	
year - Float	

Types

AdjustedTeamMetricsStddevPopFie...

aggregate stddevPop on columns

Field Name	Description
epa - Float	
epaAllowed - Float	
explosiveness - Float	
explosivenessAllowed - Float	
highlightYards - Float	
highlightYardsAllowed - Float	
lineYards - Float	
lineYardsAllowed - Float	
openFieldYards - Float	

EXAMPLE

```
{  
  "epa": 987.65,  
  "epaAllowed": 987.65,  
  "explosiveness": 123.45,  
  "explosivenessAllowed": 123.45,  
  "highlightYards": 987.65,  
  "highlightYardsAllowed": 987.65,  
  "lineYards": 123.45,  
  "lineYardsAllowed": 123.45,  
  "openFieldYards": 123.45,  
  "openFieldYardsAllowed": 987.65,  
  "passingDownsSuccess": 123.45,  
  "passingDownsSuccessAllowed": 123  
  "passingEpa": 987.65,  
  "passingEpaAllowed": 123.45,  
  "rushingEpa": 987.65,  
  "rushingEpaAllowed": 123.45,  
  "secondLevelYards": 987.65,  
  "secondLevelYardsAllowed": 123.45  
  "standardDownsSuccess": 123.45,  
  "standardDownsSuccessAllowed": 12  
  "success": 987.65,  
  "successAllowed": 987.65,  
  "teamId": 987.65,  
  "year": 123.45  
}
```

Field Name	Description
openFieldYardsAllowed	- Float
passingDownsSuccess	- Float
passingDownsSuccessAllowed	- Float
passingEpa	- Float
passingEpaAllowed	- Float
rushingEpa	- Float
rushingEpaAllowed	- Float
secondLevelYards	- Float
secondLevelYardsAllowed	- Float
standardDownsSuccess	- Float
standardDownsSuccessAllowed	- Float
success	- Float

Field Name	Description
successAllowed -	
Float	
teamId -	Float
year -	Float

Types

AdjustedTeamMetricsStddevSampF...

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
epa -	Float
epaAllowed -	Float
explosiveness -	
Float	
explosivenessAllowed	
-	Float

```
{
  "epa": 123.45,
  "epaAllowed": 987.65,
  "explosiveness": 987.65,
  "explosivenessAllowed": 123.45,
  "highlightYards": 987.65,
  "highlightYardsAllowed": 987.65,
  "lineYards": 987.65,
  "lineYardsAllowed": 987.65,
  "openFieldYards": 987.65,
  "openFieldYardsAllowed": 987.65,
  "passingDownsSuccess": 123.45,
  "passingDownsSuccessAllowed": 123
  "passingEpa": 123.45,
  "passingEpaAllowed": 987.65,
```

Field Name	Description
highlightYards -	
Float	
highlightYardsAllowed	
- Float	
lineYards -	
Float	
lineYardsAllowed	
- Float	
openFieldYards -	
Float	
openFieldYardsAllowed	
- Float	
passingDownsSuccess	
- Float	
passingDownsSuccessAllowed	
- Float	
passingEpa -	
Float	
passingEpaAllowed	
- Float	
rushingEpa -	
Float	
rushingEpaAllowed	
- Float	

```
"rushingEpa": 987.65,  
"rushingEpaAllowed": 987.65,  
"secondLevelYards": 123.45,  
"secondLevelYardsAllowed": 123.45  
"standardDownsSuccess": 987.65,  
"standardDownsSuccessAllowed": 12  
"success": 123.45,  
"successAllowed": 123.45,  
"teamId": 987.65,  
"year": 123.45  
}
```

Field Name	Description
secondLevelYards	- Float
secondLevelYardsAllowed	- Float
standardDownsSuccess	- Float
standardDownsSuccessAllowed	- Float
success	- Float
successAllowed	- Float
teamId	- Float
year	- Float

Types

AdjustedTeamMetricsSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
epa - numeric	
epaAllowed - numeric	
explosiveness - numeric	
explosivenessAllowed - numeric	
highlightYards - numeric	
highlightYardsAllowed - numeric	
lineYards - numeric	
lineYardsAllowed - numeric	
openFieldYards - numeric	
openFieldYardsAllowed - numeric	
passingDownsSuccess - numeric	

```
{
  "epa": numeric,
  "epaAllowed": numeric,
  "explosiveness": numeric,
  "explosivenessAllowed": numeric,
  "highlightYards": numeric,
  "highlightYardsAllowed": numeric,
  "lineYards": numeric,
  "lineYardsAllowed": numeric,
  "openFieldYards": numeric,
  "openFieldYardsAllowed": numeric,
  "passingDownsSuccess": numeric,
  "passingDownsSuccessAllowed": num
  "passingEpa": numeric,
  "passingEpaAllowed": numeric,
  "rushingEpa": numeric,
  "rushingEpaAllowed": numeric,
  "secondLevelYards": numeric,
  "secondLevelYardsAllowed": numeri
  "standardDownsSuccess": numeric,
  "standardDownsSuccessAllowed": nu
  "success": numeric,
  "successAllowed": numeric,
  "teamId": 123,
  "year": smallint
}
```

Field Name	Description
passingDownsSuccessAllowed	- numeric
passingEpa	- numeric
passingEpaAllowed	- numeric
rushingEpa	- numeric
rushingEpaAllowed	- numeric
secondLevelYards	- numeric
secondLevelYardsAllowed	- numeric
standardDownsSuccess	- numeric
standardDownsSuccessAllowed	- numeric
success	- numeric
successAllowed	- numeric
teamId	- Int

Field Name	Description
year - smallint	

Types

AdjustedTeamMetricsVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
epa - Float	
epaAllowed - Float	
explosiveness - Float	
explosivenessAllowed - Float	
highlightYards - Float	
highlightYardsAllowed - Float	

```
{
  "epa": 123.45,
  "epaAllowed": 123.45,
  "explosiveness": 123.45,
  "explosivenessAllowed": 123.45,
  "highlightYards": 987.65,
  "highlightYardsAllowed": 987.65,
  "lineYards": 987.65,
  "lineYardsAllowed": 987.65,
  "openFieldYards": 123.45,
  "openFieldYardsAllowed": 987.65,
  "passingDownsSuccess": 987.65,
  "passingDownsSuccessAllowed": 123.45,
  "passingEpa": 123.45,
  "passingEpaAllowed": 123.45,
  "rushingEpa": 987.65,
  "rushingEpaAllowed": 123.45,
  "secondLevelYards": 987.65,
  "secondLevelYardsAllowed": 123.45,
  "standardDownsSuccess": 123.45,
```

Field Name	Description	
lineYards - Float		"standardDownsSuccessAllowed": 98
lineYardsAllowed - Float		"success": 987.65,
openFieldYards - Float		"successAllowed": 123.45,
openFieldYardsAllowed - Float		"teamId": 987.65,
passingDownsSuccess - Float		"year": 987.65
passingDownsSuccessAllowed - Float		}
passingEpa - Float		
passingEpaAllowed - Float		
rushingEpa - Float		
rushingEpaAllowed - Float		
secondLevelYards - Float		
secondLevelYardsAllowed - Float		

Field Name	Description
standardDownsSuccess	
- Float	
standardDownsSuccessAllowed	
- Float	
success - Float	
successAllowed - Float	
teamId - Float	
year - Float	

Types

AdjustedTeamMetricsVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
epa - Float	
epaAllowed - Float	

```
{
  "epa": 123.45,
  "epaAllowed": 987.65,
  "explosiveness": 987.65,
  "explosivenessAllowed": 123.45,
  "highlightYards": 123.45,
```

Field Name	Description
explosiveness -	
Float	
explosivenessAllowed	
- Float	
highlightYards -	
Float	
highlightYardsAllowed	
- Float	
lineYards -	Float
lineYardsAllowed	
- Float	
openFieldYards -	
Float	
openFieldYardsAllowed	
- Float	
passingDownsSuccess	
- Float	
passingDownsSuccessAllowed	
- Float	
passingEpa -	Float
passingEpaAllowed	
- Float	

```
"highlightYardsAllowed": 123.45,
"lineYards": 987.65,
"lineYardsAllowed": 123.45,
"openFieldYards": 987.65,
"openFieldYardsAllowed": 123.45,
"passingDownsSuccess": 987.65,
"passingDownsSuccessAllowed": 123.45,
"passingEpa": 123.45,
"passingEpaAllowed": 123.45,
"rushingEpa": 987.65,
"rushingEpaAllowed": 987.65,
"secondLevelYards": 123.45,
"secondLevelYardsAllowed": 123.45,
"standardDownsSuccess": 987.65,
"standardDownsSuccessAllowed": 123.45,
"success": 123.45,
"successAllowed": 987.65,
"teamId": 123.45,
"year": 987.65
}
```

Field Name	Description
rushingEpa - Float	
rushingEpaAllowed - Float	
secondLevelYards - Float	
secondLevelYardsAllowed - Float	
standardDownsSuccess - Float	
standardDownsSuccessAllowed - Float	
success - Float	
successAllowed - Float	
teamId - Float	
year - Float	

Types

AdjustedTeamMetricsVarianceFields

aggregate variance on columns

Field Name	Description
epa - Float	
epaAllowed - Float	
explosiveness - Float	
explosivenessAllowed - Float	
highlightYards - Float	
highlightYardsAllowed - Float	
lineYards - Float	
lineYardsAllowed - Float	
openFieldYards - Float	
openFieldYardsAllowed - Float	

EXAMPLE

```
{
  "epa": 123.45,
  "epaAllowed": 123.45,
  "explosiveness": 123.45,
  "explosivenessAllowed": 987.65,
  "highlightYards": 987.65,
  "highlightYardsAllowed": 123.45,
  "lineYards": 123.45,
  "lineYardsAllowed": 987.65,
  "openFieldYards": 123.45,
  "openFieldYardsAllowed": 987.65,
  "passingDownsSuccess": 123.45,
  "passingDownsSuccessAllowed": 987.65,
  "passingEpa": 987.65,
  "passingEpaAllowed": 123.45,
  "rushingEpa": 123.45,
  "rushingEpaAllowed": 987.65,
  "secondLevelYards": 123.45,
  "secondLevelYardsAllowed": 123.45,
  "standardDownsSuccess": 987.65,
  "standardDownsSuccessAllowed": 123.45,
  "success": 987.65,
  "successAllowed": 987.65,
  "teamId": 123.45,
  "year": 987.65
}
```

Field Name	Description
passingDownsSuccess	- Float
passingDownsSuccessAllowed	- Float
passingEpa	- Float
passingEpaAllowed	- Float
rushingEpa	- Float
rushingEpaAllowed	- Float
secondLevelYards	- Float
secondLevelYardsAllowed	- Float
standardDownsSuccess	- Float
standardDownsSuccessAllowed	- Float
success	- Float
successAllowed	- Float

Field Name	Description
teamId - Float	
year - Float	

Types

Athlete

columns and relationships of "athlete"

EXAMPLE

```
{
  "adjustedPlayerMetrics": [AdjustedPlayerMetrics],
  "adjustedPlayerMetricsAggregate": AdjustedPlayerMetricsAggregate,
  "athleteTeams": [AthleteTeam],
  "athleteTeamsAggregate": AthleteTeamsAggregate,
  "firstName": "abc123",
  "height": smallint,
  "hometown": Hometown,
  "hometownId": 987,
  "id": bigint,
  "jersey": smallint,
  "lastName": "abc123",
  "name": "abc123",
  "position": Position,
  "positionId": smallint,
  "recruits": [Recruit],
  "recruitsAggregate": RecruitAggregate,
  "teamId": 123,
```

Field Name	Description
adjustedPlayerMetrics - [AdjustedPlayerMetrics!]!	An array relationship

Arguments

distinctOn - [\[AdjustedPlayerMetricsSelectColumn!\]](#)

distinct select on columns

limit - [Int](#)

Field Name	Description	
limit	limit the number of rows returned	"weight": smallint }
offset - Int		
skip the first n rows. Use only with order_by		
orderBy - AdjustedPlayerMetricsOrderBy!		
sort the rows by one or more columns		
where - AdjustedPlayerMetricsBoolExp		
filter the rows returned		
adjustedPlayerMetricsAggregate	An aggregate	
-	relationship	
AdjustedPlayerMetricsAggregate!		
Arguments		
distinctOn - AdjustedPlayerMetricsSelectColumn!		
distinct select on columns		
limit - Int		
limit the number of rows returned		
offset - Int		

Field Name	Description
------------	-------------

skip -	skip the first n rows. Use only with order_by
--------	--

orderBy -	[AdjustedPlayerMetricsOrderBy!]
-----------	---------------------------------

sort the rows by one or more columns	
--------------------------------------	--

where -	AdjustedPlayerMetricsBoolExp
---------	------------------------------

filter the rows returned	
--------------------------	--

athleteTeams -	An array
[AthleteTeam!]!	relationship

Arguments	
-----------	--

distinctOn -	[AthleteTeamSelectColumn!]
--------------	----------------------------

distinct select on columns	
----------------------------	--

limit -	Int
---------	-----

limit the number of rows returned	
-----------------------------------	--

offset -	Int
----------	-----

skip the first n rows. Use only with order_by	
--	--

orderBy -	[AthleteTeamOrderBy!]
-----------	-----------------------

sort the rows by one or more columns	
--------------------------------------	--

Field Name	Description
------------	-------------

where - AthleteTeamBoolExp	
--	--

filter the rows returned

athleteTeamsAggregate	An aggregate
-----------------------	--------------

-
relationship
[AthleteTeamAggregate!](#)

Arguments	
-----------	--

distinctOn -	[AthleteTeamSelectColumn!]
--------------	--

distinct select on columns

limit - Int	
-----------------------------	--

limit the number of rows returned

offset - Int	
------------------------------	--

skip the first n rows. Use only with
order_by

orderBy - [AthleteTeamOrderBy!]	
---	--

sort the rows by one or more columns

where - AthleteTeamBoolExp	
--	--

filter the rows returned

firstName - String	
------------------------------------	--

Field Name	Description
height - smallint	
hometown - Hometown	An object relationship
hometownId - Int	
id - bigint!	
jersey - smallint	
lastName - String	
name - String!	
position - Position	An object relationship
positionId - smallint	
recruits - [Recruit!]!	An array relationship

Arguments

distinctOn - [\[RecruitSelectColumn!\]](#)

distinct select on columns

limit - [Int](#)

limit the number of rows returned

offset - [Int](#)

Field Name	Description
------------	-------------

skip - Int	skip the first n rows. Use only with order_by
------------	---

orderBy - [RecruitOrderBy!]	sort the rows by one or more columns
-----------------------------	--------------------------------------

where - RecruitBoolExp	filter the rows returned
------------------------	--------------------------

recruitsAggregate	An aggregate relationship
-	RecruitAggregate!

Arguments	
-----------	--

distinctOn - [RecruitSelectColumn!]	
-------------------------------------	--

distinct select on columns	
----------------------------	--

limit - Int	
-------------	--

limit the number of rows returned	
-----------------------------------	--

offset - Int	
--------------	--

skip the first n rows. Use only with order_by	
---	--

orderBy - [RecruitOrderBy!]	
-----------------------------	--

sort the rows by one or more columns	
--------------------------------------	--

Field Name	Description
------------	-------------

where - <code>RecruitBoolExp</code>	
-------------------------------------	--

filter the rows returned

teamId - <code>Int</code>	
---------------------------	--

weight - <code>smallint</code>	
--------------------------------	--

Types

AthleteAggregate

aggregated selection of "athlete"

EXAMPLE

```
{  
  "aggregate": AthleteAggregateFiel  
  "nodes": [Athlete]  
}
```

Field Name	Description
------------	-------------

aggregate -	
-------------	--

<code>AthleteAggregateFields</code>	
-------------------------------------	--

nodes -	
---------	--

<code>[Athlete!]!</code>	
--------------------------	--

Types

AthleteAggregateBoolExp

Input Field	Description	EXAMPLE
count - athleteAggregateBoolExpCount		{ "count": athleteAggregateBoolExpCo [REDACTED]

Types

AthleteAggregateFields

aggregate fields of "athlete"

EXAMPLE

Field Name	Description
avg - AthleteAvgFields	
count - Int!	

Arguments

{
 "avg": AthleteAvgFields,
 "count": 123,
 "max": AthleteMaxFields,
 "min": AthleteMinFields,
 "stddev": AthleteStddevFields,
 "stddevPop": AthleteStddevPopFiel
 "stddevSamp": AthleteStddevSampFi
 "sum": AthleteSumFields,
 "varPop": AthleteVarPopFields,
 "varSamp": AthleteVarSampFields,

Field Name	Description	
columns - [AthleteSelectColumn!]		"variance": AthleteVarianceFields
distinct - Boolean		}
max -		
	AthleteMaxFields	
min -		
	AthleteMinFields	
stddev -		
	AthleteStddevFields	
stddevPop -		
	AthleteStddevPopFields	
stddevSamp -		
	AthleteStddevSampFields	
sum -		
	AthleteSumFields	
varPop -		
	AthleteVarPopFields	
varSamp -		
	AthleteVarSampFields	
variance -		
	AthleteVarianceFields	

Types

AthleteAggregateOrderBy

order by aggregate values of table
"athlete"

Input Field	Description
avg -	AthleteAvgOrderBy
count - OrderBy	
max -	AthleteMaxOrderBy
min -	AthleteMinOrderBy
stddev -	AthleteStddevOrderBy
stddevPop -	AthleteStddevPopOrderBy
stddevSamp -	AthleteStddevSampOrderBy
sum -	AthleteSumOrderBy

EXAMPLE

```
{  
  "avg": AthleteAvgOrderBy,  
  "count": "ASC",  
  "max": AthleteMaxOrderBy,  
  "min": AthleteMinOrderBy,  
  "stddev": AthleteStddevOrderBy,  
  "stddevPop": AthleteStddevPopOrderBy  
  "stddevSamp": AthleteStddevSampOrderBy,  
  "sum": AthleteSumOrderBy,  
  "varPop": AthleteVarPopOrderBy,  
  "varSamp": AthleteVarSampOrderBy,  
  "variance": AthleteVarianceOrderBy  
}
```

Input Field	Description
varPop -	
	AthleteVarPopOrderBy
varSamp -	
	AthleteVarSampOrderBy
variance -	
	AthleteVarianceOrderBy

Types

AthleteAvgFields

aggregate avg on columns

EXAMPLE

```
{
  "height": 987.65,
  "hometownId": 123.45,
  "id": 123.45,
  "jersey": 987.65,
  "positionId": 987.65,
  "teamId": 987.65,
  "weight": 987.65
}
```

Field Name	Description
height - Float	
hometownId - Float	
id - Float	
jersey - Float	
positionId - Float	

Field Name	Description
teamId - Float	
weight - Float	

Types

AthleteAvgOrderBy

order by avg() on columns of table
"athlete"

EXAMPLE

```
{  
  "height": "ASC",  
  "hometownId": "ASC",  
  "id": "ASC",  
  "jersey": "ASC",  
  "positionId": "ASC",  
  "teamId": "ASC",  
  "weight": "ASC"  
}
```

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	

Input Field	Description
weight - OrderBy	

Types

AthleteBoolExp

Boolean expression to filter rows from the table "athlete". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[AthleteBoolExp!]
_not -	AthleteBoolExp
_or -	[AthleteBoolExp!]
adjustedPlayerMetrics	-
	AdjustedPlayerMetricsBoolExp

EXAMPLE

```
{
  "_and": [AthleteBoolExp],
  "_not": AthleteBoolExp,
  "_or": [AthleteBoolExp],
  "adjustedPlayerMetrics": Adjusted
  "adjustedPlayerMetricsAggregate": AthleteTeamBoolExp
  "athletes": AthleteBoolExp
  "athletesAggregate": AthleteTeamBoolExp
  "firstName": StringComparisonExp,
  "height": SmallintComparisonExp,
  "hometown": HometownBoolExp,
  "hometownId": IntComparisonExp,
  "id": BigintComparisonExp,
  "jersey": SmallintComparisonExp,
  "lastName": StringComparisonExp,
  "name": StringComparisonExp,
  "position": PositionBoolExp,
  "positionId": SmallintComparisonExp
  "recruits": RecruitBoolExp,
  "recruitsAggregate": RecruitAggregate
}
```

Input Field	Description	
adjustedPlayerMetricsAggregate	-	"teamId": IntComparisonExp, "weight": SmallintComparisonExp }
	AdjustedPlayerMetricsAggregateBoolExp	
athleteTeams	-	
	AthleteTeamBoolExp	
athleteTeamsAggregate	-	
	AthleteTeamAggregateBoolExp	
firstName	-	
	StringComparisonExp	
height	-	
	SmallintComparisonExp	
hometown	-	
	HometownBoolExp	
hometownId	-	
	IntComparisonExp	
id	-	
	BigintComparisonExp	
jersey	-	
	SmallintComparisonExp	
lastName	-	
	StringComparisonExp	

Input Field	Description
name -	StringComparisonExp
position -	PositionBoolExp
positionId -	SmallintComparisonExp
recruits -	RecruitBoolExp
recruitsAggregate	-
	RecruitAggregateBoolExp
teamId -	IntComparisonExp
weight -	SmallintComparisonExp

Types

AthleteMaxFields

aggregate max on columns

Field Name	Description
firstName - <code>String</code>	
height - <code>smallint</code>	
hometownId - <code>Int</code>	
id - <code>bigint</code>	
jersey - <code>smallint</code>	
lastName - <code>String</code>	
name - <code>String</code>	
positionId - <code>smallint</code>	
teamId - <code>Int</code>	
weight - <code>smallint</code>	

EXAMPLE

```
{  
  "firstName": "xyz789",  
  "height": smallint,  
  "hometownId": 987,  
  "id": bigint,  
  "jersey": smallint,  
  "lastName": "abc123",  
  "name": "abc123",  
  "positionId": smallint,  
  "teamId": 123,  
  "weight": smallint  
}
```

Types

AthleteMaxOrderBy

order by max() on columns of table
"athlete"

Input Field	Description
firstName -	
OrderBy	
height -	OrderBy
hometownId -	
OrderBy	
id -	OrderBy
jersey -	OrderBy
lastName -	OrderBy
name -	OrderBy
positionId -	
OrderBy	
teamId -	OrderBy
weight -	OrderBy

EXAMPLE

```
{
  "firstName": "ASC",
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "lastName": "ASC",
  "name": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
  "weight": "ASC"
}
```

Types

AthleteMinFields

aggregate min on columns

Field Name	Description
firstName - <code>String</code>	
height - <code>smallint</code>	
hometownId - <code>Int</code>	
id - <code>bigint</code>	
jersey - <code>smallint</code>	
lastName - <code>String</code>	
name - <code>String</code>	
positionId - <code>smallint</code>	
teamId - <code>Int</code>	
weight - <code>smallint</code>	

EXAMPLE

```
{  
  "firstName": "xyz789",  
  "height": smallint,  
  "hometownId": 123,  
  "id": bigint,  
  "jersey": smallint,  
  "lastName": "abc123",  
  "name": "abc123",  
  "positionId": smallint,  
  "teamId": 123,  
  "weight": smallint  
}
```

Types

AthleteMinOrderBy

order by min() on columns of table
"athlete"

Input Field	Description
firstName -	
height -	OrderBy
hometownId -	
id -	OrderBy
jersey -	OrderBy
lastName -	OrderBy
name -	OrderBy
positionId -	
teamId -	OrderBy
weight -	OrderBy

EXAMPLE

```
{  
  "firstName": "ASC",  
  "height": "ASC",  
  "hometownId": "ASC",  
  "id": "ASC",  
  "jersey": "ASC",  
  "lastName": "ASC",  
  "name": "ASC",  
  "positionId": "ASC",  
  "teamId": "ASC",  
  "weight": "ASC"  
}
```

Types

AthleteOrderBy

Ordering options when selecting data from "athlete".

Input Field	Description
adjustedPlayerMetricsAggregate	-
	AdjustedPlayerMetricsAggregateOrderBy
athleteTeamsAggregate	-
	AthleteTeamAggregateOrderBy
firstName	-
	OrderBy
height	- OrderBy
hometown	-
	HometownOrderBy
hometownId	-
	OrderBy
id	- OrderBy
jersey	- OrderBy

EXAMPLE

```
{  
  "adjustedPlayerMetricsAggregate":  
    "athleteTeamsAggregate": AthleteT  
    "firstName": "ASC",  
    "height": "ASC",  
    "hometown": HometownOrderBy,  
    "hometownId": "ASC",  
    "id": "ASC",  
    "jersey": "ASC",  
    "lastName": "ASC",  
    "name": "ASC",  
    "position": PositionOrderBy,  
    "positionId": "ASC",  
    "recruitsAggregate": RecruitAggre  
    "teamId": "ASC",  
    "weight": "ASC"  
}
```

Input Field	Description
lastName - OrderBy	
name - OrderBy	
position - PositionOrderBy	
positionId - OrderBy	
recruitsAggregate	
-	
RecruitAggregateOrderBy	
teamId - OrderBy	
weight - OrderBy	

Types

AthleteSelectColumn

select columns of table "athlete"

EXAMPLE

["firstName"](#)

Enum Value	Description
firstName	column name
height	column name
hometownId	column name
id	column name
jersey	column name
lastName	column name
name	column name
positionId	column name
teamId	column name
weight	column name

Types

AthleteStddevFields

aggregate stddev on columns

EXAMPLE

```
{  
  "height": 123.45,  
  "weight": 180.5,
```

Field Name	Description	
height - Float		"hometownId": 123.45, "id": 123.45, "jersey": 987.65, "positionId": 123.45, "teamId": 123.45, "weight": 123.45
hometownId - Float		}
id - Float		
jersey - Float		
positionId - Float		
teamId - Float		
weight - Float		

Types

AthleteStddevOrderBy

order by stddev() on columns of table
"athlete"

Input Field	Description
height - OrderBy	

EXAMPLE

```
{
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
```

Input Field	Description	
hometownId -		"weight": "ASC"
OrderBy		}
id - OrderBy		
jersey - OrderBy		
positionId -		
OrderBy		
teamId - OrderBy		
weight - OrderBy		

Types

AthleteStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
height - Float	
hometownId - Float	

```
{
  "height": 123.45,
  "hometownId": 987.65,
  "id": 123.45,
  "jersey": 123.45,
  "positionId": 123.45,
  "teamId": 123.45,
```

Field Name	Description	
id - Float		"weight": 123.45
jersey - Float		}
positionId - Float		
teamId - Float		
weight - Float		

Types

AthleteStddevPopOrderBy

order by stddevPop() on columns of
table "athlete"

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	

EXAMPLE

```
{
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
  "weight": "ASC"
}
```

Input Field	Description
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	
weight - OrderBy	

Types

AthleteStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
height - Float	
hometownId - Float	
id - Float	
jersey - Float	
positionId - Float	

```
{
  "height": 123.45,
  "hometownId": 123.45,
  "id": 987.65,
  "jersey": 987.65,
  "positionId": 123.45,
  "teamId": 123.45,
  "weight": 987.65
}
```

Field Name	Description
teamId - Float	
weight - Float	

Types

AthleteStddevSampOrderBy

order by stddevSamp() on columns of
table "athlete"

EXAMPLE

```
{
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
  "weight": "ASC"
}
```

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	

Input Field	Description
weight - OrderBy	

Types

AthleteSumFields

aggregate sum on columns

Field Name	Description
height - smallint	
hometownId - Int	
id - bigint	
jersey - smallint	
positionId - smallint	
teamId - Int	
weight - smallint	

EXAMPLE

```
{  
  "height": smallint,  
  "hometownId": 123,  
  "id": bigint,  
  "jersey": smallint,  
  "positionId": smallint,  
  "teamId": 987,  
  "weight": smallint  
}
```

Types

AthleteSumOrderBy

order by sum() on columns of table
"athlete"

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	
weight - OrderBy	

EXAMPLE

```
{  
  "height": "ASC",  
  "hometownId": "ASC",  
  "id": "ASC",  
  "jersey": "ASC",  
  "positionId": "ASC",  
  "teamId": "ASC",  
  "weight": "ASC"  
}
```

Types

AthleteTeam

columns and relationships of
"athlete_team"

Field Name	Description
athlete - Athlete	An object relationship
athleteId - bigint!	
endYear - smallint	
startYear - smallint	
team - historicalTeam	An object relationship
teamId - Int!	

EXAMPLE

```
{  
  "athlete": Athlete,  
  "athleteId": bigint,  
  "endYear": smallint,  
  "startYear": smallint,  
  "team": historicalTeam,  
  "teamId": 987  
}
```

Types

AthleteTeamAggregate

aggregated selection of "athlete_team"

Field Name	Description
aggregate -	
AthleteTeamAggregateFields	
nodes -	
[AthleteTeam!]!	

EXAMPLE

```
{  
  "aggregate": AthleteTeamAggregate  
  "nodes": [AthleteTeam]  
}
```

Types

AthleteTeamAggregateBoolExp

Input Field	Description
count -	
athleteTeamAggregateBoolExpCount	

EXAMPLE

```
{"count": athleteTeamAggregateBoolE
```

Types

AthleteTeamAggregateFields

aggregate fields of "athlete_team"

Field Name	Description
avg -	AthleteTeamAvgFields
count - Int!	<div><p>Arguments</p><p>columns - [AthleteTeamSelectColumn!]</p><p>distinct - Boolean</p></div>
max -	AthleteTeamMaxFields
min -	AthleteTeamMinFields
stddev -	AthleteTeamStddevFields
stddevPop -	AthleteTeamStddevPopFields
stddevSamp -	AthleteTeamStddevSampFields

EXAMPLE

```
{  
  "avg": AthleteTeamAvgFields,  
  "count": 123,  
  "max": AthleteTeamMaxFields,  
  "min": AthleteTeamMinFields,  
  "stddev": AthleteTeamStddevFields  
  "stddevPop": AthleteTeamStddevPop  
  "stddevSamp": AthleteTeamStddevSa  
  "sum": AthleteTeamSumFields,  
  "varPop": AthleteTeamVarPopFields  
  "varSamp": AthleteTeamVarSampFiel  
  "variance": AthleteTeamVarianceFi  
}
```

Field Name	Description
sum -	AthleteTeamSumFields
varPop -	AthleteTeamVarPopFields
varSamp -	AthleteTeamVarSampFields
variance -	AthleteTeamVarianceFields

Types

AthleteTeamAggregateOrderBy

order by aggregate values of table
"athlete_team"

EXAMPLE

```
{
  "avg": AthleteTeamAvgOrderBy,
  "count": "ASC",
  "max": AthleteTeamMaxOrderBy,
  "min": AthleteTeamMinOrderBy,
  "stddev": AthleteTeamStddevOrderByB,
  "stddevPop": AthleteTeamStddevPop,
  "stddevSamp": AthleteTeamStddevSa,
  "sum": AthleteTeamSumOrderBy,
```

Input Field	Description
avg -	AthleteTeamAvgOrderBy
count - OrderBy	

Input Field	Description
max -	AthleteTeamMaxOrderBy
min -	AthleteTeamMinOrderBy
stddev -	AthleteTeamStddevOrderBy
stddevPop -	AthleteTeamStddevPopOrderBy
stddevSamp -	AthleteTeamStddevSampOrderBy
sum -	AthleteTeamSumOrderBy
varPop -	AthleteTeamVarPopOrderBy
varSamp -	AthleteTeamVarSampOrderBy
variance -	AthleteTeamVarianceOrderBy

```
"varPop": AthleteTeamVarPopOrderBy  
"varSamp": AthleteTeamVarSampOrderBy  
"variance": AthleteTeamVarianceOrderBy  
}
```

Types

AthleteTeamAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
athleteId - Float	
endYear - Float	
startYear - Float	
teamId - Float	

```
{"athleteId": 123.45, "endYear": 98
```

Types

AthleteTeamAvgOrderBy

order by avg() on columns of table
"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy

Types

AthleteTeamBoolExp

Boolean expression to filter rows from the table "athlete_team". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	
[AthleteTeamBoolExp!]	
_not -	
AthleteTeamBoolExp	

EXAMPLE

```
{
  "_and": [AthleteTeamBoolExp],
  "_not": AthleteTeamBoolExp,
  "_or": [AthleteTeamBoolExp],
  "athlete": AthleteBoolExp,
  "athleteId": BigintComparisonExp,
  "endYear": SmallintComparisonExp,
  "startYear": SmallintComparisonExp
  "team": historicalTeamBoolExp,
```

Input Field	Description	
_or -		"teamId": IntComparisonExp
	[AthleteTeamBoolExp!]	}
athlete -		
	AthleteBoolExp	
athleteId -		
	BigintComparisonExp	
endYear -		
	SmallintComparisonExp	
startYear -		
	SmallintComparisonExp	
team -		
	historicalTeamBoolExp	
teamId -		
	IntComparisonExp	

Types

AthleteTeamMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
athleteId - bigint	
endYear - smallint	
startYear - smallint	
teamId - Int	

```
{
  "athleteId": bigint,
  "endYear": smallint,
  "startYear": smallint,
  "teamId": 123
}
```

Types

AthleteTeamMaxOrderBy

order by max() on columns of table

EXAMPLE

"athlete_team"

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId - OrderBy	
endYear - OrderBy	

Input Field	Description
startYear -	
teamId - OrderBy	

Types

AthleteTeamMinFields

aggregate min on columns

EXAMPLE

```
{  
  "athleteId": bigint,  
  "endYear": smallint,  
  "startYear": smallint,  
  "teamId": 123  
}
```

Field Name	Description
athleteId - bigint	
endYear - smallint	
startYear -	
teamId - Int	

Types

AthleteTeamMinOrderBy

order by min() on columns of table
"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy



Types

AthleteTeamOrderBy

Ordering options when selecting data from "athlete_team".

Input Field	Description
athlete -	
AthleteOrderBy	
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	OrderBy
team -	
historicalTeamOrderBy	
teamId -	OrderBy

Types

EXAMPLE

```
{
  "athlete": AthleteOrderBy,
  "athleteId": "ASC",
  "endYear": "ASC",
  "startYear": "ASC",
  "team": historicalTeamOrderBy,
  "teamId": "ASC"
}
```

AthleteTeamSelectColumn

select columns of table "athlete_team"

EXAMPLE

Enum Value	Description	
athleteId	column name	"athleteId"
endYear	column name	
startYear	column name	
teamId	column name	

Types

AthleteTeamStddevFields

aggregate stddev on columns

EXAMPLE

```
{"athleteId": 987.65, "endYear": 98
```

Field Name	Description
athleteId - Float	
endYear - Float	
startYear - Float	
teamId - Float	

Types

AthleteTeamStddevOrderBy

order by stddev() on columns of table
"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy

Types

AthleteTeamStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description	
athleteId - Float		{"athleteId": 987.65, "endYear": 12}
endYear - Float		
startYear - Float		
teamId - Float		

Types

AthleteTeamStddevPopOrderBy

order by stddevPop() on columns of
table "athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId - OrderBy	
endYear - OrderBy	
startYear - OrderBy	

Input Field	Description
teamId - OrderBy	

Types

AthleteTeamStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

```
{"athleteId": 123.45, "endYear": 98
```

Field Name	Description
athleteId - Float	
endYear - Float	
startYear - Float	
teamId - Float	

Types

AthleteTeamStddevSampOrderBy

order by stddevSamp() on columns of
table "athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy

Types

AthleteTeamSumFields

aggregate sum on columns

EXAMPLE

```
{  
  "athleteId": bigint,  
  "teamId": bigint,  
  "startYear": int,  
  "endYear": int,  
  "count": int,  
  "sum": float}
```

Field Name	Description	
athleteId - bigint		"endYear": smallint, "startYear": smallint, "teamId": 123 }
endYear - smallint		
startYear - smallint		
teamId - Int		

Types

AthleteTeamSumOrderBy

order by sum() on columns of table

"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId - OrderBy	
endYear - OrderBy	
startYear - OrderBy	

Input Field	Description
teamId - OrderBy	

Types

AthleteTeamVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
athleteId - Float	
endYear - Float	
startYear - Float	
teamId - Float	

{"athleteId": 987.65, "endYear": 98

Types

AthleteTeamVarPopOrderBy

order by varPop() on columns of table
"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy

Types

AthleteTeamVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
athleteId - Float	<code>{"athleteId": 123.45, "endYear": 12</code>
endYear - Float	
startYear - Float	
teamId - Float	

Types

AthleteTeamVarSampOrderBy

order by varSamp() on columns of table
"athlete_team"

EXAMPLE

```
{ "athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId - OrderBy	
endYear - OrderBy	
startYear - OrderBy	
teamId - OrderBy	

Types

AthleteTeamVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
athleteId - Float	
endYear - Float	
startYear - Float	
teamId - Float	

```
{"athleteId": 987.65, "endYear": 98
```

Types

AthleteTeamVarianceOrderBy

order by variance() on columns of table
"athlete_team"

EXAMPLE

```
{"athleteId": "ASC", "endYear": "AS
```

Input Field	Description
athleteId -	
OrderBy	
endYear -	OrderBy
startYear -	
OrderBy	
teamId -	OrderBy

Types

AthleteVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
height -	Float
hometownId -	Float

```
{
  "height": 123.45,
  "hometownId": 987.65,
  "id": 987.65,
  "jersey": 987.65,
  "positionId": 987.65,
  "teamId": 987.65,
```

Field Name	Description	
<code>id - Float</code>		<code>"weight": 123.45</code>
<code>jersey - Float</code>		<code>}</code>
<code>positionId - Float</code>		
<code>teamId - Float</code>		
<code>weight - Float</code>		

Types

AthleteVarPopOrderBy

order by varPop() on columns of table
 "athlete"

Input Field	Description
<code>height - OrderBy</code>	
<code>hometownId - OrderBy</code>	
<code>id - OrderBy</code>	

EXAMPLE

```
{
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
  "weight": "ASC"
}
```

Input Field	Description
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	
weight - OrderBy	

Types

AthleteVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
height - Float	
hometownId - Float	
id - Float	
jersey - Float	
positionId - Float	

```
{
  "height": 987.65,
  "hometownId": 987.65,
  "id": 987.65,
  "jersey": 123.45,
  "positionId": 987.65,
  "teamId": 123.45,
  "weight": 123.45
}
```

Field Name	Description
teamId - Float	
weight - Float	

Types

AthleteVarSampOrderBy

order by varSamp() on columns of table
"athlete"

EXAMPLE

```
{
  "height": "ASC",
  "hometownId": "ASC",
  "id": "ASC",
  "jersey": "ASC",
  "positionId": "ASC",
  "teamId": "ASC",
  "weight": "ASC"
}
```

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	

Input Field	Description
weight - OrderBy	

Types

AthleteVarianceFields

aggregate variance on columns

EXAMPLE

```
{  
  "height": 987.65,  
  "hometownId": 123.45,  
  "id": 123.45,  
  "jersey": 123.45,  
  "positionId": 987.65,  
  "teamId": 987.65,  
  "weight": 123.45  
}
```

Field Name	Description
height - Float	
hometownId - Float	
id - Float	
jersey - Float	
positionId - Float	
teamId - Float	
weight - Float	

Types

AthleteVarianceOrderBy

order by variance() on columns of table
"athlete"

Input Field	Description
height - OrderBy	
hometownId - OrderBy	
id - OrderBy	
jersey - OrderBy	
positionId - OrderBy	
teamId - OrderBy	
weight - OrderBy	

EXAMPLE

```
{  
  "height": "ASC",  
  "hometownId": "ASC",  
  "id": "ASC",  
  "jersey": "ASC",  
  "positionId": "ASC",  
  "teamId": "ASC",  
  "weight": "ASC"  
}
```

Types

BigintComparisonExp

Boolean expression to compare columns of type "bigint". All fields are combined with logical 'AND'.

Input Field	Description
_eq - <code>bigint</code>	
_gt - <code>bigint</code>	
_gte - <code>bigint</code>	
_in - <code>[bigint!]</code>	
_isNull - <code>Boolean</code>	
_lt - <code>bigint</code>	
_lte - <code>bigint</code>	
_neq - <code>bigint</code>	
_nin - <code>[bigint!]</code>	

EXAMPLE

```
{  
  "_eq": bigint,  
  "_gt": bigint,  
  "_gte": bigint,  
  "_in": [bigint],  
  "_isNull": false,  
  "_lt": bigint,  
  "_lte": bigint,  
  "_neq": bigint,  
  "_nin": [bigint]  
}
```

Types

Boolean

Types

BooleanComparisonExp

Boolean expression to compare columns of type "Boolean". All fields are combined with logical 'AND'.

Input Field	Description
_eq - Boolean	
_gt - Boolean	
_gte - Boolean	
_in - [Boolean!]	
_isNull - Boolean	
_lt - Boolean	
_lte - Boolean	
_neq - Boolean	

EXAMPLE

```
{  
  "_eq": false,  
  "_gt": true,  
  "_gte": false,  
  "_in": [true],  
  "_isNull": true,  
  "_lt": true,  
  "_lte": false,  
  "_neq": true,  
  "_nin": [false]  
}
```

Input Field	Description
_nin - [Boolean!]	

Types

Calendar

columns and relationships of "calendar"

Field Name	Description
endDate -	
timestamp!	
seasonType -	
season_type!	
startDate -	
timestamp!	
week -	smallint!
year -	smallint!

EXAMPLE

```
{  
  "endDate": timestamp,  
  "seasonType": season_type,  
  "startDate": timestamp,  
  "week": smallint,  
  "year": smallint  
}
```

Types

CalendarBoolExp

Boolean expression to filter rows from the table "calendar". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[CalendarBoolExp!]
_not -	CalendarBoolExp
_or -	[CalendarBoolExp!]
endDate -	TimestampComparisonExp
seasonType -	SeasonTypeComparisonExp
startDate -	TimestampComparisonExp
week -	SmallintComparisonExp

EXAMPLE

```
{  
  "_and": [CalendarBoolExp],  
  "_not": CalendarBoolExp,  
  "_or": [CalendarBoolExp],  
  "endDate": TimestampComparisonExp  
  "seasonType": SeasonTypeCompariso  
  "startDate": TimestampComparisonE  
  "week": SmallintCompariso  
  "year": SmallintCompariso  
}
```

Input Field	Description
year - SmallIntComparisonExp	

Types

CalendarOrderBy

Ordering options when selecting data from "calendar".

Input Field	Description
endDate - OrderBy	
seasonType - OrderBy	
startDate - OrderBy	
week - OrderBy	
year - OrderBy	

EXAMPLE

```
{  
  "endDate": "ASC",  
  "seasonType": "ASC",  
  "startDate": "ASC",  
  "week": "ASC",  
  "year": "ASC"  
}
```

Types

CalendarSelectColumn

select columns of table "calendar"

EXAMPLE

Enum Value	Description	
endDate	column name	"endDate"
seasonType	column name	
startDate	column name	
week	column name	
year	column name	

Types

Coach

columns and relationships of "coach"

EXAMPLE

Field Name	Description
firstName - String!	
id - Int!	
lastName - String!	
seasons - [CoachSeason!]!	An array relationship

Arguments

distinctOn - [CoachSeasonSelectColumn!]

distinct select on columns

limit - Int

limit the number of rows returned

offset - Int

skip the first n rows. Use only with order_by

orderBy - [CoachSeasonOrderBy!]

sort the rows by one or more columns

where - CoachSeasonBoolExp

filter the rows returned

```
{  
  "firstName": "abc123",  
  "id": 987,  
  "lastName": "xyz789",  
  "seasons": [CoachSeason],  
  "seasonsAggregate": CoachSeasonAg  
}
```

Field Name	Description
seasonsAggregate -	An aggregate relationship CoachSeasonAggregate!

Arguments

distinctOn -
[\[CoachSeasonSelectColumn!\]](#)

distinct select on columns

limit - [Int](#)

limit the number of rows returned

offset - [Int](#)

skip the first n rows. Use only with
order_by

orderBy - [\[CoachSeasonOrderBy!\]](#)

sort the rows by one or more columns

where - [CoachSeasonBoolExp](#)

filter the rows returned

Types

CoachAggregate

aggregated selection of "coach"

Field Name	Description
aggregate - CoachAggregateFields	
nodes - [Coach!]!	

EXAMPLE

```
{
  "aggregate": CoachAggregateFields
  "nodes": [Coach]
}
```

Types

CoachAggregateFields

aggregate fields of "coach"

Field Name	Description
avg - CoachAvgFields	
count - Int!	

EXAMPLE

```
{
  "avg": CoachAvgFields,
  "count": 987,
  "max": CoachMaxFields,
  "min": CoachMinFields,
  "stddev": CoachStddevFields,
  "stddevPop": CoachStddevPopFields
  "stddevSamp": CoachStddevSampFiel
  "sum": CoachSumFields,
```

Field Name	Description
Arguments	
columns - [CoachSelectColumn!]	
distinct - Boolean	
max -	CoachMaxFields
min -	CoachMinFields
stddev -	CoachStddevFields
stddevPop -	CoachStddevPopFields
stddevSamp -	CoachStddevSampFields
sum -	CoachSumFields
varPop -	CoachVarPopFields
varSamp -	CoachVarSampFields
variance -	CoachVarianceFields

```
"varPop": CoachVarPopFields,  
"varSamp": CoachVarSampFields,  
"variance": CoachVarianceFields  
}
```

Types

CoachAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
id - Float	

```
{"id": 123.45}
```

Types

CoachBoolExp

Boolean expression to filter rows from the table "coach". All fields are combined with a logical 'AND'.

EXAMPLE

Input Field	Description
_and - [CoachBoolExp!]	

```
{
  "_and": [CoachBoolExp],
  "_not": CoachBoolExp,
  "_or": [CoachBoolExp],
  "firstName": StringComparisonExp,
  "id": IntComparisonExp,
  "lastName": StringComparisonExp,
  "seasons": CoachSeasonBoolExp,
```

Input Field	Description	
_not -		"seasonsAggregate": CoachSeasonAg
CoachBoolExp		}
_or -		
[CoachBoolExp!]		
firstName -		
StringComparisonExp		
id -		
IntComparisonExp		
lastName -		
StringComparisonExp		
seasons -		
CoachSeasonBoolExp		
seasonsAggregate		
-		
CoachSeasonAggregateBoolExp		

Types

CoachMaxFields

aggregate max on columns

Field Name	Description
firstName - <code>String</code>	
id - <code>Int</code>	
lastName - <code>String</code>	

EXAMPLE

```
{  
  "firstName": "xyz789",  
  "id": 987,  
  "lastName": "abc123"  
}
```

Types

CoachMinFields

aggregate min on columns

Field Name	Description
firstName - <code>String</code>	
id - <code>Int</code>	
lastName - <code>String</code>	

EXAMPLE

```
{  
  "firstName": "abc123",  
  "id": 987,  
  "lastName": "abc123"  
}
```

Types

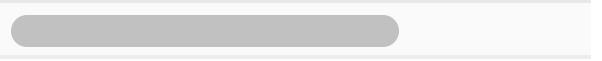
CoachOrderBy

Ordering options when selecting data from "coach".

Input Field	Description
firstName -	
OrderBy	
id -	OrderBy
lastName -	OrderBy
seasonsAggregate	
-	
	CoachSeasonAggregateOrderBy

EXAMPLE

```
{  
    "firstName": "ASC",  
    "id": "ASC",  
    "lastName": "ASC",  
    "seasonsAggregate": CoachSeasonAg  
}
```



Types

CoachSeason

columns and relationships of
"coach_season"

Field Name	Description
coach - Coach!	An object relationship
games - smallint!	
losses - smallint!	
postseasonRank - smallint	
preseasonRank - smallint	
team - currentTeams	An object relationship
ties - smallint!	
wins - smallint!	
year - smallint!	

EXAMPLE

```
{  
  "coach": Coach,  
  "games": smallint,  
  "losses": smallint,  
  "postseasonRank": smallint,  
  "preseasonRank": smallint,  
  "team": currentTeams,  
  "ties": smallint,  
  "wins": smallint,  
  "year": smallint  
}
```

Types

CoachSeasonAggregate

aggregated selection of "coach_season"

EXAMPLE

Field Name	Description
aggregate -	CoachSeasonAggregateFields
nodes -	[CoachSeason!]!

```
{  
  "aggregate": CoachSeasonAggregate  
  "nodes": [CoachSeason]  
}
```

Types

CoachSeasonAggregateBoolExp

Input Field	Description
count -	coachSeasonAggregateBoolExpCount

EXAMPLE

```
{"count": coachSeasonAggregateBoolE
```

Types

CoachSeasonAggregateFields

aggregate fields of "coach_season"

Field Name	Description
avg -	CoachSeasonAvgFields
count - Int!	
	Arguments
	columns - [CoachSeasonSelectColumn!]
	distinct - Boolean
max -	CoachSeasonMaxFields
min -	CoachSeasonMinFields
stddev -	CoachSeasonStddevFields
stddevPop -	CoachSeasonStddevPopFields

EXAMPLE

```
{  
  "avg": CoachSeasonAvgFields,  
  "count": 123,  
  "max": CoachSeasonMaxFields,  
  "min": CoachSeasonMinFields,  
  "stddev": CoachSeasonStddevFields  
  "stddevPop": CoachSeasonStddevPop  
  "stddevSamp": CoachSeasonStddevSa  
  "sum": CoachSeasonSumFields,  
  "varPop": CoachSeasonVarPopFields  
  "varSamp": CoachSeasonVarSampFiel  
  "variance": CoachSeasonVarianceFi  
}
```

Field Name	Description
stddevSamp -	CoachSeasonStddevSampFields
sum -	CoachSeasonSumFields
varPop -	CoachSeasonVarPopFields
varSamp -	CoachSeasonVarSampFields
variance -	CoachSeasonVarianceFields

Types

CoachSeasonAggregateOrderBy

order by aggregate values of table
"coach_season"

EXAMPLE

```
{  
  "avg": CoachSeasonAvgOrderBy,  
  "count": "ASC",  
  "max": CoachSeasonMaxOrderBy,  
  "min": CoachSeasonMinOrderBy,
```

Input Field	Description
avg -	
	CoachSeasonAvgOrderBy
count - OrderBy	
max -	
	CoachSeasonMaxOrderBy
min -	
	CoachSeasonMinOrderBy
stddev -	
	CoachSeasonStddevOrderBy
stddevPop -	
	CoachSeasonStddevPopOrderBy
stddevSamp -	
	CoachSeasonStddevSampOrderBy
sum -	
	CoachSeasonSumOrderBy
varPop -	
	CoachSeasonVarPopOrderBy
varSamp -	
	CoachSeasonVarSampOrderBy
variance -	
	CoachSeasonVarianceOrderBy

```

    "stddev": CoachSeasonStddevOrderByB
    "stddevPop": CoachSeasonStddevPop
    "stddevSamp": CoachSeasonStddevSa
    "sum": CoachSeasonSumOrderBy,
    "varPop": CoachSeasonVarPopOrderByB
    "varSamp": CoachSeasonVarSampOrde
    "variance": CoachSeasonVarianceOr
}

```

Types

CoachSeasonAvgFields

aggregate avg on columns

Field Name	Description
games - Float	
losses - Float	
postseasonRank - Float	
preseasonRank - Float	
ties - Float	
wins - Float	
year - Float	

EXAMPLE

```
{  
  "games": 123.45,  
  "losses": 123.45,  
  "postseasonRank": 987.65,  
  "preseasonRank": 987.65,  
  "ties": 123.45,  
  "wins": 123.45,  
  "year": 987.65  
}
```

Types

CoachSeasonAvgOrderBy

order by avg() on columns of table
"coach_season"

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

EXAMPLE

```
{  
  "games": "ASC",  
  "losses": "ASC",  
  "postseasonRank": "ASC",  
  "preseasonRank": "ASC",  
  "ties": "ASC",  
  "wins": "ASC",  
  "year": "ASC"  
}
```

Types

CoachSeasonBoolExp

Boolean expression to filter rows from the table "coach_season". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[CoachSeasonBoolExp!]
_not -	CoachSeasonBoolExp
_or -	[CoachSeasonBoolExp!]
coach -	CoachBoolExp
games -	SmallintComparisonExp
losses -	SmallintComparisonExp
postseasonRank -	SmallintComparisonExp
preseasonRank -	SmallintComparisonExp
team -	currentTeamsBoolExp

EXAMPLE

```
{  
  "_and": [CoachSeasonBoolExp],  
  "_not": CoachSeasonBoolExp,  
  "_or": [CoachSeasonBoolExp],  
  "coach": CoachBoolExp,  
  "games": SmallintComparisonExp,  
  "losses": SmallintComparisonExp,  
  "postseasonRank": SmallintComparisonExp,  
  "preseasonRank": SmallintComparisonExp,  
  "team": currentTeamsBoolExp,  
  "ties": SmallintComparisonExp,  
  "wins": SmallintComparisonExp,  
  "year": SmallintComparisonExp  
}
```

Input Field	Description
ties -	
	SmallintComparisonExp
wins -	
	SmallintComparisonExp
year -	
	SmallintComparisonExp

Types

CoachSeasonMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
games - smallint	
losses - smallint	
postseasonRank -	
smallint	

```
{
  "games": smallint,
  "losses": smallint,
  "postseasonRank": smallint,
  "preseasonRank": smallint,
  "ties": smallint,
  "wins": smallint,
  "year": smallint
}
```

Field Name	Description
preseasonRank -	
<small>smallint</small>	
ties -	<small>smallint</small>
wins -	<small>smallint</small>
year -	<small>smallint</small>

Types

CoachSeasonMaxOrderBy

order by max() on columns of table
 "coach_season"

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
games -	<small>OrderBy</small>
losses -	<small>OrderBy</small>
postseasonRank -	
<small>OrderBy</small>	

Input Field	Description
preseasonRank -	
OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
games - smallint	
losses - smallint	
postseasonRank -	
smallint	
preseasonRank -	
smallint	

```
{
  "games": smallint,
  "losses": smallint,
  "postseasonRank": smallint,
  "preseasonRank": smallint,
  "ties": smallint,
  "wins": smallint,
  "year": smallint
}
```

Field Name	Description
ties - smallint	
wins - smallint	
year - smallint	

Types

CoachSeasonMinOrderBy

order by min() on columns of table
"coach_season"

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonOrderBy

Ordering options when selecting data from "coach_season".

EXAMPLE

```
{
  "coach": CoachOrderBy,
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "team": currentTeamsOrderBy,
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
coach - CoachOrderBy	
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	

Input Field	Description
team -	
	currentTeamsOrderBy
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonSelectColumn

select columns of table "coach_season"

EXAMPLE

["games"](#)

Enum Value	Description
games	column name
losses	column name
postseasonRank	column name
preseasonRank	column name
ties	column name

Enum Value	Description
wins	column name
year	column name

Types

CoachSeasonStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
games - Float	
losses - Float	
postseasonRank - Float	
preseasonRank - Float	
ties - Float	
wins - Float	

```
{
  "games": 123.45,
  "losses": 987.65,
  "postseasonRank": 987.65,
  "preseasonRank": 123.45,
  "ties": 123.45,
  "wins": 123.45,
  "year": 123.45
}
```

Field Name	Description
year - Float	

Types

CoachSeasonStddevOrderBy

order by stddev() on columns of table
 "coach_season"

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Types

CoachSeasonStddevPopFields

aggregate stddevPop on columns

Field Name	Description
games - Float	
losses - Float	
postseasonRank - Float	
preseasonRank - Float	
ties - Float	
wins - Float	
year - Float	

EXAMPLE

```
{  
  "games": 123.45,  
  "losses": 123.45,  
  "postseasonRank": 987.65,  
  "preseasonRank": 123.45,  
  "ties": 987.65,  
  "wins": 123.45,  
  "year": 987.65  
}
```

Types

CoachSeasonStddevPopOrderBy

order by stddevPop() on columns of
table "coach_season"

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

EXAMPLE

```
{  
  "games": "ASC",  
  "losses": "ASC",  
  "postseasonRank": "ASC",  
  "preseasonRank": "ASC",  
  "ties": "ASC",  
  "wins": "ASC",  
  "year": "ASC"  
}
```

Types

CoachSeasonStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
games - Float	
losses - Float	
postseasonRank - Float	
preseasonRank - Float	
ties - Float	
wins - Float	
year - Float	

```
{  
  "games": 987.65,  
  "losses": 123.45,  
  "postseasonRank": 987.65,  
  "preseasonRank": 987.65,  
  "ties": 123.45,  
  "wins": 123.45,  
  "year": 123.45  
}
```

Types

CoachSeasonStddevSampOrderBy

order by stddevSamp() on columns of
table "coach_season"

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
games - <code>smallint</code>	
losses - <code>smallint</code>	
postseasonRank - <code>smallint</code>	
preseasonRank - <code>smallint</code>	
ties - <code>smallint</code>	
wins - <code>smallint</code>	
year - <code>smallint</code>	

```
{
  "games": smallint,
  "losses": smallint,
  "postseasonRank": smallint,
  "preseasonRank": smallint,
  "ties": smallint,
  "wins": smallint,
  "year": smallint
}
```

Types

CoachSeasonSumOrderBy

order by sum() on columns of table
"coach_season"

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
```

Input Field	Description
games - <code>OrderBy</code>	

Input Field	Description	
losses - OrderBy		"wins": "ASC", "year": "ASC"
postseasonRank - OrderBy		}
preseasonRank - OrderBy		
ties - OrderBy		
wins - OrderBy		
year - OrderBy		

Types

CoachSeasonVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
games - Float	
losses - Float	

```
{
  "games": 123.45,
  "losses": 123.45,
  "postseasonRank": 987.65,
  "preseasonRank": 987.65,
  "ties": 987.65,
  "wins": 987.65,
```

Field Name	Description	
postseasonRank -		"year": 987.65
Float		}
preseasonRank -		
Float		
ties - Float		
wins - Float		
year - Float		

Types

CoachSeasonVarPopOrderBy

order by varPop() on columns of table
 "coach_season"

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
games - OrderBy	
losses - OrderBy	

Input Field	Description
postseasonRank -	
OrderBy	
preseasonRank -	
OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
games - Float	
losses - Float	
postseasonRank -	
Float	

```
{
  "games": 123.45,
  "losses": 987.65,
  "postseasonRank": 123.45,
  "preseasonRank": 123.45,
  "ties": 123.45,
  "wins": 123.45,
  "year": 123.45
}
```

Field Name	Description
preseasonRank -	
Float	
ties - Float	
wins - Float	
year - Float	

Types

CoachSeasonVarSampOrderBy

order by varSamp() on columns of table
 "coach_season"

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank -	
OrderBy	

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
preseasonRank -	
OrderBy	
ties - OrderBy	
wins - OrderBy	
year - OrderBy	

Types

CoachSeasonVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
games - Float	
losses - Float	
postseasonRank -	
Float	
preseasonRank -	
Float	

```
{  
    "games": 987.65,  
    "losses": 123.45,  
    "postseasonRank": 987.65,  
    "preseasonRank": 987.65,  
    "ties": 987.65,  
    "wins": 123.45,  
    "year": 987.65  
}
```

Field Name	Description
ties - Float	
wins - Float	
year - Float	

Types

CoachSeasonVarianceOrderBy

order by variance() on columns of table
"coach_season"

EXAMPLE

```
{
  "games": "ASC",
  "losses": "ASC",
  "postseasonRank": "ASC",
  "preseasonRank": "ASC",
  "ties": "ASC",
  "wins": "ASC",
  "year": "ASC"
}
```

Input Field	Description
games - OrderBy	
losses - OrderBy	
postseasonRank - OrderBy	
preseasonRank - OrderBy	
ties - OrderBy	

Input Field	Description
wins - OrderBy	
year - OrderBy	

Types

CoachSelectColumn

select columns of table "coach"

EXAMPLE

`"firstName"`

Enum Value	Description
firstName	column name
id	column name
lastName	column name

Types

CoachStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

CoachStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

CoachStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name

Description

{"id": 987.65}

id - [Float](#)

Types

CoachSumFields

aggregate sum on columns

EXAMPLE

Field Name

Description

{"id": 123}

id - [Int](#)

Types

CoachVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

CoachVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 987.65}

Types

CoachVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
------------	-------------

<code>id</code> - <code>Float</code>	<code>{"id": 987.65}</code>
--------------------------------------	-----------------------------

Types

Conference

columns and relationships of "conference"

EXAMPLE

Field Name	Description
------------	-------------

<code>abbreviation</code> - <code>String</code>	{ <code>"abbreviation": "abc123",</code> <code>"division": division,</code> <code>"id": smallint,</code> <code>"name": "xyz789",</code> <code>"shortName": "abc123",</code> <code>"srName": "abc123"</code> }
---	--

<code>division</code> - <code>division</code>	
---	--

<code>id</code> - <code>smallint!</code>	
--	--

Field Name	Description
name - <code>String!</code>	
shortName - <code>String</code>	
srName - <code>String</code>	

Types

ConferenceBoolExp

Boolean expression to filter rows from the table "conference". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	<code>[ConferenceBoolExp!]</code>
_not -	<code>ConferenceBoolExp</code>
_or -	<code>[ConferenceBoolExp!]</code>

EXAMPLE

```
{
  "_and": [ConferenceBoolExp],
  "_not": ConferenceBoolExp,
  "_or": [ConferenceBoolExp],
  "abbreviation": StringComparisonE
  "division": DivisionComparisonExp
  "id": SmallintComparisonExp,
  "name": StringComparisonExp,
  "shortName": StringComparisonExp,
  "srName": StringComparisonExp
}
```



Input Field	Description
abbreviation -	StringComparisonExp
division -	DivisionComparisonExp
id -	SmallintComparisonExp
name -	StringComparisonExp
shortName -	StringComparisonExp
srName -	StringComparisonExp

Types

ConferenceOrderBy

Ordering options when selecting data from "conference".

EXAMPLE

```
{  
  "abbreviation": "ASC",
```

Input Field	Description	
abbreviation - OrderBy		"division": "ASC", "id": "ASC", "name": "ASC", "shortName": "ASC", "srName": "ASC"
division - OrderBy		}
id - OrderBy		
name - OrderBy		
shortName - OrderBy		
srName - OrderBy		

Types

ConferenceSelectColumn

select columns of table "conference"

EXAMPLE

Enum Value	Description	
abbreviation	column name	"abbreviation"
division	column name	

Enum Value	Description
id	column name
name	column name
shortName	column name
srName	column name

Types

DivisionComparisonExp

Boolean expression to compare columns of type "division". All fields are combined with logical 'AND'.

EXAMPLE

```
{
  "_eq": division,
  "_gt": division,
  "_gte": division,
  "_in": [division],
  "_isNull": true,
  "_lt": division,
  "_lte": division,
  "_neq": division,
  "_nin": [division]
}
```

Input Field	Description
_eq - division	
_gt - division	
_gte - division	
_in - [division!]	

Input Field	Description
_isNull - Boolean	
_lt - division	
_lte - division	
_neq - division	
_nin - [division!]	

Types

DraftPicks

columns and relationships of
"draft_picks"

Field Name	Description
collegeAthleteRecord - Athlete	An object relationship
collegeId - Int	
collegeTeam - historicalTeam	An object relationship

EXAMPLE

```
{
  "collegeAthleteRecord": Athlete,
  "collegeId": 123,
  "collegeTeam": historicalTeam,
  "collegeTeamId": 123,
  "draftTeam": DraftTeam,
  "grade": smallint,
  "height": smallint,
  "name": "abc123",
  "nflTeamId": smallint,
  "overall": smallint,
  "overallRank": smallint,
```

Field Name	Description
collegeTeamId - Int!	
draftTeam - DraftTeam!	An object relationship
grade - smallint	
height - smallint	
name - String!	
nflTeamId - smallint!	
overall - smallint!	
overallRank - smallint	
pick - smallint!	
position - DraftPosition!	An object relationship
positionId - smallint!	
positionRank - smallint	
round - smallint!	

```

    "pick": smallint,
    "position": DraftPosition,
    "positionId": smallint,
    "positionRank": smallint,
    "round": smallint,
    "weight": smallint,
    "year": smallint
}

```

Field Name	Description
weight - smallint	
year - smallint!	

Types

DraftPicksAggregate

aggregated selection of "draft_picks"

EXAMPLE

```
{  
  "aggregate": DraftPicksAggregateF  
  "nodes": [DraftPicks]  
}
```

Field Name	Description
aggregate - DraftPicksAggregateFields	
nodes - [DraftPicks!]!	

Types

DraftPicksAggregateBoolExp

Input Field	Description	EXAMPLE
count - DraftPicksAggregateBoolExpCount		{"count": draftPicksAggregateBoolEx

Types

DraftPicksAggregateFields

aggregate fields of "draft_picks"

EXAMPLE

Field Name	Description
avg - DraftPicksAvgFields	
count - Int! Arguments columns - [DraftPicksSelectColumn!]	

```
{
  "avg": DraftPicksAvgFields,
  "count": 123,
  "max": DraftPicksMaxFields,
  "min": DraftPicksMinFields,
  "stddev": DraftPicksStddevFields,
  "stddevPop": DraftPicksStddevPopF
  "stddevSamp": DraftPicksStddevSam
  "sum": DraftPicksSumFields,
  "varPop": DraftPicksVarPopFields,
  "varSamp": DraftPicksVarSampField
```

Field Name	Description	
distinct - Boolean		"variance": DraftPicksVarianceFie }
max -		
	DraftPicksMaxFields	
min -		
	DraftPicksMinFields	
stddev -		
	DraftPicksStddevFields	
stddevPop -		
	DraftPicksStddevPopFields	
stddevSamp -		
	DraftPicksStddevSampFields	
sum -		
	DraftPicksSumFields	
varPop -		
	DraftPicksVarPopFields	
varSamp -		
	DraftPicksVarSampFields	
variance -		
	DraftPicksVarianceFields	

Types

DraftPicksAggregateOrderBy

order by aggregate values of table
"draft_picks"

Input Field	Description
avg -	DraftPicksAvgOrderBy
count - OrderBy	
max -	DraftPicksMaxOrderBy
min -	DraftPicksMinOrderBy
stddev -	DraftPicksStddevOrderBy
stddevPop -	DraftPicksStddevPopOrderBy
stddevSamp -	DraftPicksStddevSampOrderBy
sum -	DraftPicksSumOrderBy

EXAMPLE

```
{  
  "avg": DraftPicksAvgOrderBy,  
  "count": "ASC",  
  "max": DraftPicksMaxOrderBy,  
  "min": DraftPicksMinOrderBy,  
  "stddev": DraftPicksStddevOrderBy  
  "stddevPop": DraftPicksStddevPop0  
  "stddevSamp": DraftPicksStddevSam  
  "sum": DraftPicksSumOrderBy,  
  "varPop": DraftPicksVarPopOrderBy  
  "varSamp": DraftPicksVarSampOrderBy  
  "variance": DraftPicksVarianceOrd  
}
```

Input Field	Description
varPop -	
	DraftPicksVarPopOrderBy
varSamp -	
	DraftPicksVarSampOrderBy
variance -	
	DraftPicksVarianceOrderBy

Types

DraftPicksAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
collegeId - Float	
collegeTeamId -	
Float	
grade - Float	
height - Float	

```
{
  "collegeId": 987.65,
  "collegeTeamId": 123.45,
  "grade": 987.65,
  "height": 987.65,
  "nflTeamId": 123.45,
  "overall": 123.45,
  "overallRank": 987.65,
  "pick": 987.65,
  "positionId": 123.45,
  "positionRank": 123.45,
  "round": 987.65,
```

Field Name	Description	
nflTeamId - Float		"weight": 987.65, "year": 987.65 }
overall - Float		
overallRank - Float		
pick - Float		
positionId - Float		
positionRank - Float		
round - Float		
weight - Float		
year - Float		

Types

DraftPicksAvgOrderBy

order by avg() on columns of table
"draft_picks"

EXAMPLE

```
{  
  "collegeId": "ASC",
```

Input Field	Description
collegeId - OrderBy	"collegeTeamId": "ASC", "grade": "ASC", "height": "ASC", "nflTeamId": "ASC", "overall": "ASC", "overallRank": "ASC", "pick": "ASC", "positionId": "ASC", "positionRank": "ASC", "round": "ASC", "weight": "ASC", "year": "ASC"
collegeTeamId - OrderBy	
grade - OrderBy	
height - OrderBy	
nflTeamId - OrderBy	
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	
positionId - OrderBy	
positionRank - OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksBoolExp

Boolean expression to filter rows from the table "draft_picks". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[DraftPicksBoolExp!]
_not -	DraftPicksBoolExp
_or -	[DraftPicksBoolExp!]
collegeAthleteRecord	- AthleteBoolExp
collegeId -	IntComparisonExp
collegeTeam -	historicalTeamBoolExp
collegeTeamId -	IntComparisonExp

EXAMPLE

```
{  
  "_and": [DraftPicksBoolExp],  
  "_not": DraftPicksBoolExp,  
  "_or": [DraftPicksBoolExp],  
  "collegeAthleteRecord": AthleteBo  
  "collegeId": IntComparisonExp,  
  "collegeTeam": historicalTeamBool  
  "collegeTeamId": IntComparisonExp  
  "draftTeam": DraftTeamBoolExp,  
  "grade": SmallintComparisonExp,  
  "height": SmallintComparisonExp,  
  "name": StringComparisonExp,  
  "nflTeamId": SmallintComparisonEx  
  "overall": SmallintComparisonExp,  
  "overallRank": SmallintCompariso  
  "pick": SmallintComparisonExp,  
  "position": DraftPositionBoolExp,  
  "positionId": SmallintCompariso  
  "positionRank": SmallintCompariso  
  "round": SmallintComparisonExp,  
  "weight": SmallintComparisonExp,  
  "year": SmallintComparisonExp  
}
```

Input Field	Description
draftTeam -	DraftTeamBoolExp
grade -	SmallintComparisonExp
height -	SmallintComparisonExp
name -	StringComparisonExp
nflTeamId -	SmallintComparisonExp
overall -	SmallintComparisonExp
overallRank -	SmallintComparisonExp
pick -	SmallintComparisonExp
position -	DraftPositionBoolExp
positionId -	SmallintComparisonExp
positionRank -	SmallintComparisonExp

Input Field	Description
round -	
	SmallintComparisonExp
weight -	
	SmallintComparisonExp
year -	
	SmallintComparisonExp

Types

DraftPicksMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
collegeId - Int	
collegeTeamId -	
Int	
grade - smallint	
height - smallint	

```
{
  "collegeId": 123,
  "collegeTeamId": 123,
  "grade": smallint,
  "height": smallint,
  "name": "xyz789",
  "nflTeamId": smallint,
  "overall": smallint,
  "overallRank": smallint,
  "pick": smallint,
  "positionId": smallint,
  "positionRank": smallint,
```

Field Name	Description	
name - <code>String</code>		<code>"round": smallint,</code>
nflTeamId - <code>smallint</code>		<code>"weight": smallint,</code>
overall - <code>smallint</code>		<code>"year": smallint</code>
overallRank - <code>smallint</code>		
pick - <code>smallint</code>		
positionId - <code>smallint</code>		
positionRank - <code>smallint</code>		
round - <code>smallint</code>		
weight - <code>smallint</code>		
year - <code>smallint</code>		

Types

DraftPicksMaxOrderBy

order by max() on columns of table
 "draft_picks"

Input Field	Description
collegeId -	
OrderBy	
collegeTeamId -	
OrderBy	
grade - OrderBy	
height - OrderBy	
name - OrderBy	
nflTeamId -	
OrderBy	
overall - OrderBy	
overallRank -	
OrderBy	
pick - OrderBy	
positionId -	
OrderBy	
positionRank -	
OrderBy	
round - OrderBy	

EXAMPLE

```
{
  "collegeId": "ASC",
  "collegeTeamId": "ASC",
  "grade": "ASC",
  "height": "ASC",
  "name": "ASC",
  "nflTeamId": "ASC",
  "overall": "ASC",
  "overallRank": "ASC",
  "pick": "ASC",
  "positionId": "ASC",
  "positionRank": "ASC",
  "round": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Input Field	Description
weight - OrderBy	
year - OrderBy	

Types

DraftPicksMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
collegeId - Int	
collegeTeamId - Int	
grade - smallint	
height - smallint	
name - String	
nflTeamId - smallint	

```
{
  "collegeId": 123,
  "collegeTeamId": 123,
  "grade": smallint,
  "height": smallint,
  "name": "abc123",
  "nflTeamId": smallint,
  "overall": smallint,
  "overallRank": smallint,
  "pick": smallint,
  "positionId": smallint,
  "positionRank": smallint,
  "round": smallint,
  "weight": smallint,
  "year": smallint
}
```

Field Name	Description
overall - <code>smallint</code>	
overallRank - <code>smallint</code>	
pick - <code>smallint</code>	
positionId - <code>smallint</code>	
positionRank - <code>smallint</code>	
round - <code>smallint</code>	
weight - <code>smallint</code>	
year - <code>smallint</code>	

Types

DraftPicksMinOrderBy

order by min() on columns of table
"draft_picks"

EXAMPLE

```
{  
  "collegeId": "ASC",
```

Input Field	Description
collegeId - OrderBy	"collegeTeamId": "ASC", "grade": "ASC", "height": "ASC", "name": "ASC", "nflTeamId": "ASC", "overall": "ASC", "overallRank": "ASC", "pick": "ASC", "positionId": "ASC", "positionRank": "ASC", "round": "ASC", "weight": "ASC", "year": "ASC"
collegeTeamId - OrderBy	}
grade - OrderBy	
height - OrderBy	
name - OrderBy	
nflTeamId - OrderBy	
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	
positionId - OrderBy	
positionRank - OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksOrderBy

Ordering options when selecting data from "draft_picks".

Input Field	Description
collegeAthleteRecord	- AthleteOrderBy
collegeId	- OrderBy
collegeTeam	- historicalTeamOrderBy
collegeTeamId	- OrderBy
draftTeam	- DraftTeamOrderBy
grade	- OrderBy
height	- OrderBy
name	- OrderBy

EXAMPLE

```
{  
  "collegeAthleteRecord": AthleteOr  
  "collegeId": "ASC",  
  "collegeTeam": historicalTeamOrde  
  "collegeTeamId": "ASC",  
  "draftTeam": DraftTeamOrderBy,  
  "grade": "ASC",  
  "height": "ASC",  
  "name": "ASC",  
  "nflTeamId": "ASC",  
  "overall": "ASC",  
  "overallRank": "ASC",  
  "pick": "ASC",  
  "position": DraftPositionOrderBy,  
  "positionId": "ASC",  
  "positionRank": "ASC",  
  "round": "ASC",  
  "weight": "ASC",  
  "year": "ASC"  
}
```

Input Field	Description
nflTeamId -	
OrderBy	
overall -	OrderBy
overallRank -	
OrderBy	
pick -	OrderBy
position -	
DraftPositionOrderBy	
positionId -	
OrderBy	
positionRank -	
OrderBy	
round -	OrderBy
weight -	OrderBy
year -	OrderBy

Types

DraftPicksSelectColumn

select columns of table "draft_picks"

EXAMPLE

Enum Value	Description	"collegeId"
collegeId	column name	
collegeTeamId	column name	
grade	column name	
height	column name	
name	column name	
nflTeamId	column name	
overall	column name	
overallRank	column name	
pick	column name	
positionId	column name	
positionRank	column name	
round	column name	
weight	column name	
year	column name	

Types

DraftPicksStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
collegeId - Float	
collegeTeamId - Float	
grade - Float	
height - Float	
nflTeamId - Float	
overall - Float	
overallRank - Float	
pick - Float	
positionId - Float	
positionRank - Float	
round - Float	
weight - Float	
year - Float	

```
{  
  "collegeId": 987.65,  
  "collegeTeamId": 123.45,  
  "grade": 123.45,  
  "height": 987.65,  
  "nflTeamId": 987.65,  
  "overall": 987.65,  
  "overallRank": 987.65,  
  "pick": 987.65,  
  "positionId": 987.65,  
  "positionRank": 987.65,  
  "round": 123.45,  
  "weight": 123.45,  
  "year": 987.65  
}
```

Types

DraftPicksStddevOrderBy

order by stddev() on columns of table
"draft_picks"

Input Field	Description
collegeId - OrderBy	
collegeTeamId - OrderBy	
grade - OrderBy	
height - OrderBy	
nflTeamId - OrderBy	
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	

EXAMPLE

```
{  
  "collegeId": "ASC",  
  "collegeTeamId": "ASC",  
  "grade": "ASC",  
  "height": "ASC",  
  "nflTeamId": "ASC",  
  "overall": "ASC",  
  "overallRank": "ASC",  
  "pick": "ASC",  
  "positionId": "ASC",  
  "positionRank": "ASC",  
  "round": "ASC",  
  "weight": "ASC",  
  "year": "ASC"  
}
```

Input Field	Description
positionId -	
OrderBy	
positionRank -	
OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
collegeId - Float	
collegeTeamId -	
Float	
grade - Float	

```
{
  "collegeId": 987.65,
  "collegeTeamId": 987.65,
  "grade": 987.65,
  "height": 123.45,
  "nflTeamId": 123.45,
  "overall": 987.65,
  "overallRank": 987.65,
  "pick": 123.45,
  "positionId": 123.45,
```

Field Name	Description	
height - Float		"positionRank": 987.65,
nflTeamId - Float		"round": 123.45,
overall - Float		"weight": 123.45,
overallRank - Float		"year": 987.65
pick - Float		}
positionId - Float		
positionRank - Float		
round - Float		
weight - Float		
year - Float		

Types

DraftPicksStddevPopOrderBy

order by stddevPop() on columns of
table "draft_picks"

Input Field	Description
collegeId -	
OrderBy	
collegeTeamId -	
OrderBy	
grade -	OrderBy
height -	OrderBy
nflTeamId -	
OrderBy	
overall -	OrderBy
overallRank -	
OrderBy	
pick -	OrderBy
positionId -	
OrderBy	
positionRank -	
OrderBy	
round -	OrderBy
weight -	OrderBy

EXAMPLE

```
{
    "collegeId": "ASC",
    "collegeTeamId": "ASC",
    "grade": "ASC",
    "height": "ASC",
    "nflTeamId": "ASC",
    "overall": "ASC",
    "overallRank": "ASC",
    "pick": "ASC",
    "positionId": "ASC",
    "positionRank": "ASC",
    "round": "ASC",
    "weight": "ASC",
    "year": "ASC"
}
```

Input Field	Description
year - OrderBy	

Types

DraftPicksStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
collegeId - Float	
collegeTeamId - Float	
grade - Float	
height - Float	
nflTeamId - Float	
overall - Float	
overallRank - Float	

```
{
  "collegeId": 987.65,
  "collegeTeamId": 123.45,
  "grade": 987.65,
  "height": 987.65,
  "nflTeamId": 123.45,
  "overall": 123.45,
  "overallRank": 987.65,
  "pick": 987.65,
  "positionId": 987.65,
  "positionRank": 987.65,
  "round": 987.65,
  "weight": 123.45,
  "year": 123.45
}
```

Field Name	Description
pick - Float	
positionId - Float	
positionRank - Float	
round - Float	
weight - Float	
year - Float	

Types

DraftPicksStddevSampOrderBy

order by stddevSamp() on columns of
table "draft_picks"

EXAMPLE

```
{
  "collegeId": "ASC",
  "collegeTeamId": "ASC",
  "grade": "ASC",
  "height": "ASC",
  "nflTeamId": "ASC",
  "overall": "ASC",
  "overallRank": "ASC",
  "pick": "ASC",
```

Input Field	Description
collegeId - OrderBy	

Input Field	Description	
collegeTeamId -		"positionId": "ASC",
OrderBy		"positionRank": "ASC",
grade - OrderBy		"round": "ASC",
height - OrderBy		"weight": "ASC",
nflTeamId -		"year": "ASC"
OrderBy		}
overall - OrderBy		
overallRank -		
OrderBy		
pick - OrderBy		
positionId -		
OrderBy		
positionRank -		
OrderBy		
round - OrderBy		
weight - OrderBy		
year - OrderBy		

Types

DraftPicksSumFields

aggregate sum on columns

Field Name	Description
collegeId - <code>Int</code>	
collegeTeamId - <code>Int</code>	
grade - <code>smallint</code>	
height - <code>smallint</code>	
nflTeamId - <code>smallint</code>	
overall - <code>smallint</code>	
overallRank - <code>smallint</code>	
pick - <code>smallint</code>	
positionId - <code>smallint</code>	
positionRank - <code>smallint</code>	
round - <code>smallint</code>	

EXAMPLE

```
{  
  "collegeId": 987,  
  "collegeTeamId": 987,  
  "grade": smallint,  
  "height": smallint,  
  "nflTeamId": smallint,  
  "overall": smallint,  
  "overallRank": smallint,  
  "pick": smallint,  
  "positionId": smallint,  
  "positionRank": smallint,  
  "round": smallint,  
  "weight": smallint,  
  "year": smallint  
}
```

Field Name	Description
weight - smallint	
year - smallint	

Types

DraftPicksSumOrderBy

order by sum() on columns of table
"draft_picks"

EXAMPLE

```
{
  "collegeId": "ASC",
  "collegeTeamId": "ASC",
  "grade": "ASC",
  "height": "ASC",
  "nflTeamId": "ASC",
  "overall": "ASC",
  "overallRank": "ASC",
  "pick": "ASC",
  "positionId": "ASC",
  "positionRank": "ASC",
  "round": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Input Field	Description
collegeId - OrderBy	
collegeTeamId - OrderBy	
grade - OrderBy	
height - OrderBy	
nflTeamId - OrderBy	

Input Field	Description
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	
positionId - OrderBy	
positionRank - OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksVarPopFields

aggregate varPop on columns

EXAMPLE

```
{  
  "collegeId": 123.45,
```

Field Name	Description
collegeId - Float	"collegeTeamId": 123.45, "grade": 987.65, "height": 987.65, "nflTeamId": 987.65, "overall": 123.45, "overallRank": 123.45, "pick": 987.65, "positionId": 987.65, "positionRank": 987.65, "round": 123.45, "weight": 123.45, "year": 123.45
collegeTeamId - Float	}
grade - Float	
height - Float	
nflTeamId - Float	
overall - Float	
overallRank - Float	
pick - Float	
positionId - Float	
positionRank - Float	
round - Float	
weight - Float	
year - Float	

Types

DraftPicksVarPopOrderBy

order by varPop() on columns of table
"draft_picks"

Input Field	Description
collegeId -	
OrderBy	
collegeTeamId -	
OrderBy	
grade -	OrderBy
height -	OrderBy
nflTeamId -	
OrderBy	
overall -	OrderBy
overallRank -	
OrderBy	
pick -	OrderBy
positionId -	
OrderBy	
positionRank -	
OrderBy	

EXAMPLE

```
{  
    "collegeId": "ASC",  
    "collegeTeamId": "ASC",  
    "grade": "ASC",  
    "height": "ASC",  
    "nflTeamId": "ASC",  
    "overall": "ASC",  
    "overallRank": "ASC",  
    "pick": "ASC",  
    "positionId": "ASC",  
    "positionRank": "ASC",  
    "round": "ASC",  
    "weight": "ASC",  
    "year": "ASC"  
}
```

Input Field	Description
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
collegeId - Float	
collegeTeamId - Float	
grade - Float	
height - Float	
nflTeamId - Float	
overall - Float	

```
{
  "collegeId": 987.65,
  "collegeTeamId": 123.45,
  "grade": 987.65,
  "height": 123.45,
  "nflTeamId": 987.65,
  "overall": 123.45,
  "overallRank": 987.65,
  "pick": 123.45,
  "positionId": 123.45,
  "positionRank": 987.65,
  "round": 123.45,
  "weight": 123.45,
  "year": 123.45
}
```

Field Name	Description
overallRank -	
Float	
pick -	Float
positionId -	Float
positionRank -	
Float	
round -	Float
weight -	Float
year -	Float

Types

DraftPicksVarSampOrderBy

order by varSamp() on columns of table
"draft_picks"

EXAMPLE

```
{  
    "collegeId": "ASC",  
    "collegeTeamId": "ASC",  
    "grade": "ASC",  
    "height": "ASC",  
    "nflTeamId": "ASC",
```

Input Field	Description
collegeId - OrderBy	"overall": "ASC", "overallRank": "ASC", "pick": "ASC", "positionId": "ASC", "positionRank": "ASC", "round": "ASC", "weight": "ASC", "year": "ASC"
collegeTeamId - OrderBy	}
grade - OrderBy	
height - OrderBy	
nflTeamId - OrderBy	
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	
positionId - OrderBy	
positionRank - OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPicksVarianceFields

aggregate variance on columns

Field Name	Description
collegeId - Float	
collegeTeamId - Float	
grade - Float	
height - Float	
nflTeamId - Float	
overall - Float	
overallRank - Float	
pick - Float	
positionId - Float	
positionRank - Float	
round - Float	

EXAMPLE

```
{  
  "collegeId": 987.65,  
  "collegeTeamId": 123.45,  
  "grade": 123.45,  
  "height": 987.65,  
  "nflTeamId": 123.45,  
  "overall": 987.65,  
  "overallRank": 123.45,  
  "pick": 123.45,  
  "positionId": 123.45,  
  "positionRank": 987.65,  
  "round": 123.45,  
  "weight": 123.45,  
  "year": 987.65  
}
```

Field Name	Description
weight - Float	
year - Float	

Types

DraftPicksVarianceOrderBy

order by variance() on columns of table
"draft_picks"

EXAMPLE

```
{
  "collegeId": "ASC",
  "collegeTeamId": "ASC",
  "grade": "ASC",
  "height": "ASC",
  "nflTeamId": "ASC",
  "overall": "ASC",
  "overallRank": "ASC",
  "pick": "ASC",
  "positionId": "ASC",
  "positionRank": "ASC",
  "round": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Input Field	Description
collegeId - OrderBy	
collegeTeamId - OrderBy	
grade - OrderBy	
height - OrderBy	
nflTeamId - OrderBy	

Input Field	Description
overall - OrderBy	
overallRank - OrderBy	
pick - OrderBy	
positionId - OrderBy	
positionRank - OrderBy	
round - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

DraftPosition

columns and relationships of
"draft_position"

EXAMPLE

```
{  
  "abbreviation": "abc123",
```

Field Name	Description	
abbreviation -		
String!		
id - smallint!		
name - String!		

Types

DraftPositionBoolExp

Boolean expression to filter rows from the table "draft_position". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[DraftPositionBoolExp!]
_not -	DraftPositionBoolExp
_or -	[DraftPositionBoolExp!]

EXAMPLE

```
{
  "_and": [DraftPositionBoolExp],
  "_not": DraftPositionBoolExp,
  "_or": [DraftPositionBoolExp],
  "abbreviation": StringComparisonExp,
  "id": SmallintComparisonExp,
  "name": StringComparisonExp
}
```



Input Field	Description
abbreviation -	StringComparisonExp
id -	SmallintComparisonExp
name -	StringComparisonExp

Types

DraftPositionOrderBy

Ordering options when selecting data from "draft_position".

EXAMPLE

```
{"abbreviation": "ASC", "id": "ASC"}
```

Input Field	Description
abbreviation -	OrderBy
id -	OrderBy
name -	OrderBy

Types

DraftPositionSelectColumn

select columns of table "draft_position"

EXAMPLE

Enum Value	Description	"abbreviation"
abbreviation	column name	
id	column name	
name	column name	

Types

DraftTeam

columns and relationships of
"draft_team"

EXAMPLE

```
{  
  "displayName": "abc123",
```

Field Name	Description
displayName - <code>String</code>	
<code>id</code> - <code>smallint!</code>	
location - <code>String!</code>	
logo - <code>String</code>	
mascot - <code>String</code>	
nickname - <code>String</code>	
picks - <code>[DraftPicks!]!</code>	An array relationship

```

"id": smallint,
"location": "abc123",
"logo": "xyz789",
"mascot": "abc123",
"nickname": "xyz789",
"picks": [DraftPicks],
"picksAggregate": DraftPicksAggre
"shortDisplayName": "xyz789"
}

```

Arguments
<code>distinctOn</code> - <code>[DraftPicksSelectColumn!]</code>
distinct select on columns
<code>limit</code> - <code>Int</code>
limit the number of rows returned
<code>offset</code> - <code>Int</code>
skip the first n rows. Use only with <code>order_by</code>
<code>orderBy</code> - <code>[DraftPicksOrderBy!]</code>
sort the rows by one or more columns

Field Name	Description
------------	-------------

where - DraftPicksBoolExp	
---	--

filter the rows returned

picksAggregate - An aggregate DraftPicksAggregate relationship	
--	--

Arguments	
-----------	--

distinctOn - [DraftPicksSelectColumn!]	
--	--

distinct select on columns

limit - Int	
-----------------------------	--

limit the number of rows returned

offset - Int	
------------------------------	--

skip the first n rows. Use only with order_by

orderBy - [DraftPicksOrderBy!]	
--	--

sort the rows by one or more columns

where - DraftPicksBoolExp	
---	--

filter the rows returned

shortDisplayName	
------------------	--

- [String](#)

Types

DraftTeamBoolExp

Boolean expression to filter rows from the table "draft_team". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[DraftTeamBoolExp!]
_not -	DraftTeamBoolExp
_or -	[DraftTeamBoolExp!]
displayName -	StringComparisonExp
id -	SmallintComparisonExp
location -	StringComparisonExp
logo -	StringComparisonExp

EXAMPLE

```
{  
  "_and": [DraftTeamBoolExp],  
  "_not": DraftTeamBoolExp,  
  "_or": [DraftTeamBoolExp],  
  "displayName": StringComparisonEx  
  "id": SmallintComparisonExp,  
  "location": StringComparisonExp,  
  "logo": StringComparisonExp,  
  "mascot": StringComparisonExp,  
  "nickname": StringComparisonExp,  
  "picks": DraftPicksBoolExp,  
  "picksAggregate": DraftPicksAggre  
  "shortDisplayName": StringComparisonExp  
}
```

Input Field	Description
mascot -	
	StringComparisonExp
nickname -	
	StringComparisonExp
picks -	
	DraftPicksBoolExp
picksAggregate -	
	DraftPicksAggregateBoolExp
shortDisplayName	
-	
	StringComparisonExp

Types

DraftTeamOrderBy

Ordering options when selecting data from "draft_team".

EXAMPLE

```
{  
  "displayName": "ASC",  
  "id": "ASC",  
  "location": "ASC",
```

Input Field	Description	
displayName - OrderBy		"logo": "ASC", "mascot": "ASC", "nickname": "ASC", "picksAggregate": DraftPicksAggre "shortDisplayName": "ASC"
id - OrderBy		}
location - OrderBy		
logo - OrderBy		
mascot - OrderBy		
nickname - OrderBy		
picksAggregate - DraftPicksAggregateOrderBy		
shortDisplayName - OrderBy		

Types

DraftTeamSelectColumn

select columns of table "draft_team"

EXAMPLE

"displayName"

Enum Value	Description
displayName	column name
id	column name
location	column name
logo	column name
mascot	column name
nickname	column name
shortDisplayName	column name

Types

Float

EXAMPLE

123.45

Types

FloatComparisonExp

Boolean expression to compare columns of type "Float". All fields are combined with logical 'AND'.

Input Field	Description
_eq - Float	
_gt - Float	
_gte - Float	
_in - [Float!]	
_isNull - Boolean	
_lt - Float	
_lte - Float	
_neq - Float	
_nin - [Float!]	

EXAMPLE

```
{  
  "_eq": 123.45,  
  "_gt": 123.45,  
  "_gte": 987.65,  
  "_in": [987.65],  
  "_isNull": false,  
  "_lt": 123.45,  
  "_lte": 123.45,  
  "_neq": 987.65,  
  "_nin": [987.65]  
}
```

Types

GameLines

columns and relationships of
"game_lines"

Field Name	Description
gameId - <code>Int!</code>	
linesProviderId - <code>Int!</code>	
moneylineAway - <code>Int</code>	
moneylineHome - <code>Int</code>	
overUnder - <code>numeric</code>	
overUnderOpen - <code>numeric</code>	
provider - <code>LinesProvider</code>	An object relationship
spread - <code>numeric</code>	
spreadOpen - <code>numeric</code>	

EXAMPLE

```
{  
  "gameId": 123,  
  "linesProviderId": 987,  
  "moneylineAway": 987,  
  "moneylineHome": 123,  
  "overUnder": numeric,  
  "overUnderOpen": numeric,  
  "provider": LinesProvider,  
  "spread": numeric,  
  "spreadOpen": numeric  
}
```

Types

GameLinesAggregate

aggregated selection of "game_lines"

EXAMPLE

Field Name	Description
aggregate -	GameLinesAggregateFields
nodes -	[GameLines!]!

```
{  
  "aggregate": GameLinesAggregateFi  
  "nodes": [GameLines]  
}
```

Types

GameLinesAggregateBoolExp

Input Field	Description	EXAMPLE
count - gameLinesAggregateBoolExpCount		{ "count": gameLinesAggregateBoolExpCount }

Types

GameLinesAggregateFields

aggregate fields of "game_lines"

Field Name	Description	EXAMPLE
avg - GameLinesAvgFields		{ "avg": GameLinesAvgFields, "count": 987, "max": GameLinesMaxFields, "min": GameLinesMinFields, "stddev": GameLinesStddevFields, "stddevPop": GameLinesStddevPopFields, "stddevSamp": GameLinesStddevSampFields, "sum": GameLinesSumFields, "varPop": GameLinesVarPopFields, "varSamp": GameLinesVarSampFields, "variance": GameLinesVarianceFields }
count - Int!		
Arguments		
columns - [GameLinesSelectColumn!]		
distinct - Boolean		
max - GameLinesMaxFields		

Field Name	Description
min -	GameLinesMinFields
stddev -	GameLinesStddevFields
stddevPop -	GameLinesStddevPopFields
stddevSamp -	GameLinesStddevSampFields
sum -	GameLinesSumFields
varPop -	GameLinesVarPopFields
varSamp -	GameLinesVarSampFields
variance -	GameLinesVarianceFields

Types

GameLinesAggregateOrderBy

order by aggregate values of table
"game_lines"

Input Field	Description
avg -	GameLinesAvgOrderBy
count - OrderBy	
max -	GameLinesMaxOrderBy
min -	GameLinesMinOrderBy
stddev -	GameLinesStddevOrderBy
stddevPop -	GameLinesStddevPopOrderBy
stddevSamp -	GameLinesStddevSampOrderBy
sum -	GameLinesSumOrderBy
varPop -	GameLinesVarPopOrderBy
varSamp -	GameLinesVarSampOrderBy

EXAMPLE

```
{
  "avg": GameLinesAvgOrderBy,
  "count": "ASC",
  "max": GameLinesMaxOrderBy,
  "min": GameLinesMinOrderBy,
  "stddev": GameLinesStddevOrderBy,
  "stddevPop": GameLinesStddevPopOrderBy,
  "stddevSamp": GameLinesStddevSampOrderBy,
  "sum": GameLinesSumOrderBy,
  "varPop": GameLinesVarPopOrderBy,
  "varSamp": GameLinesVarSampOrderBy,
  "variance": GameLinesVarianceOrderBy
}
```

Input Field	Description
variance -	GameLinesVarianceOrderBy

Types

GameLinesAvgFields

aggregate avg on columns

Field Name	Description
gameId - Float	
linesProviderId -	
Float	
moneylineAway -	
Float	
moneylineHome -	
Float	
overUnder -	
Float	

EXAMPLE

```
{  
  "gameId": 987.65,  
  "linesProviderId": 123.45,  
  "moneylineAway": 987.65,  
  "moneylineHome": 987.65,  
  "overUnder": 123.45,  
  "overUnderOpen": 123.45,  
  "spread": 123.45,  
  "spreadOpen": 123.45  
}
```

Field Name	Description
overUnder0Open -	
Float	
spread -	Float
spreadOpen -	Float

Types

GameLinesAvgOrderBy

order by avg() on columns of table
 "game_lines"

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
  "spreadOpen": "ASC"
}
```

Input Field	Description
gameId -	OrderBy
linesProviderId -	OrderBy
moneylineAway -	OrderBy

Input Field	Description
moneylineHome -	
OrderBy	
overUnder -	
OrderBy	
overUnderOpen -	
OrderBy	
spread - OrderBy	
spreadOpen -	
OrderBy	

Types

GameLinesBoolExp

Boolean expression to filter rows from the table "game_lines". All fields are combined with a logical 'AND'.

EXAMPLE

```
{  
    "_and": [GameLinesBoolExp],  
    "_not": GameLinesBoolExp,  
    "_or": [GameLinesBoolExp],  
    "gameId": IntComparisonExp,  
    "linesProviderId": IntComparisonExp  
    "moneylineAway": IntComparisonExp}
```

Input Field	Description
_and -	
[GameLinesBoolExp!]	
_not -	
GameLinesBoolExp	
_or -	
[GameLinesBoolExp!]	
gameId -	
IntComparisonExp	
linesProviderId -	
IntComparisonExp	
moneylineAway -	
IntComparisonExp	
moneylineHome -	
IntComparisonExp	
overUnder -	
NumericComparisonExp	
overUnderOpen -	
NumericComparisonExp	
provider -	
LinesProviderBoolExp	
spread -	
NumericComparisonExp	

```

"moneylineHome": IntComparisonExp
"overUnder": NumericComparisonExp
"overUnderOpen": NumericComparisonExp
"provider": LinesProviderBoolExp,
"spread": NumericComparisonExp,
"spreadOpen": NumericComparisonExp
}
}
```

Input Field	Description
spreadOpen - NumericComparisonExp	

Types

GameLinesMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
gameId - Int	
linesProviderId - Int	
moneylineAway - Int	
moneylineHome - Int	
overUnder - numeric	

```
{
  "gameId": 123,
  "linesProviderId": 987,
  "moneylineAway": 987,
  "moneylineHome": 123,
  "overUnder": numeric,
  "overUnderOpen": numeric,
  "spread": numeric,
  "spreadOpen": numeric
}
```

Field Name	Description
overUnder0open - numeric	
spread - numeric	
spreadOpen - numeric	

Types

GameLinesMaxOrderBy

order by max() on columns of table
 "game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
  "spreadOpen": "ASC"
}
```

Input Field	Description
moneylineHome -	
OrderBy	
overUnder -	
OrderBy	
overUnderOpen -	
OrderBy	
spread - OrderBy	
spreadOpen -	
OrderBy	

Types

GameLinesMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
gameId - Int	

```
{
  "gameId": 987,
  "linesProviderId": 123,
  "moneylineAway": 987,
  "moneylineHome": 987,
  "overUnder": numeric,
  "overUnderOpen": numeric,
```

Field Name	Description	
linesProviderId -		"spread": numeric, "spreadOpen": numeric
Int		}
moneylineAway -		
Int		
moneylineHome -		
Int		
overUnder -		
numeric		
overUnderOpen -		
numeric		
spread - numeric		
spreadOpen -		
numeric		

Types

GameLinesMinOrderBy

order by min() on columns of table
"game_lines"

EXAMPLE

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

```
{  
  "gameId": "ASC",  
  "linesProviderId": "ASC",  
  "moneylineAway": "ASC",  
  "moneylineHome": "ASC",  
  "overUnder": "ASC",  
  "overUnderOpen": "ASC",  
  "spread": "ASC",  
  "spreadOpen": "ASC"  
}
```

Types

GameLinesOrderBy

Ordering options when selecting data from "game_lines".

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
provider - LinesProviderOrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

EXAMPLE

```
{  
  "gameId": "ASC",  
  "linesProviderId": "ASC",  
  "moneylineAway": "ASC",  
  "moneylineHome": "ASC",  
  "overUnder": "ASC",  
  "overUnderOpen": "ASC",  
  "provider": LinesProviderOrderBy,  
  "spread": "ASC",  
  "spreadOpen": "ASC"  
}
```

Types

GameLinesSelectColumn

select columns of table "game_lines"

EXAMPLE

Enum Value	Description	"gameId"
gameId	column name	
linesProviderId	column name	
moneylineAway	column name	
moneylineHome	column name	
overUnder	column name	
overUnderOpen	column name	
spread	column name	
spreadOpen	column name	

Types

GameLinesStddevFields

aggregate stddev on columns

Field Name	Description
gameId - Float	
linesProviderId - Float	
moneylineAway - Float	
moneylineHome - Float	
overUnder - Float	
overUnderOpen - Float	
spread - Float	
spreadOpen - Float	

EXAMPLE

```
{  
  "gameId": 123.45,  
  "linesProviderId": 123.45,  
  "moneylineAway": 987.65,  
  "moneylineHome": 123.45,  
  "overUnder": 987.65,  
  "overUnderOpen": 123.45,  
  "spread": 123.45,  
  "spreadOpen": 123.45  
}
```

Types

GameLinesStddevOrderBy

order by stddev() on columns of table
 "game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
  "spreadOpen": "ASC"
}
```

Types

GameLinesStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
gameId - Float	
linesProviderId - Float	
moneylineAway - Float	
moneylineHome - Float	
overUnder - Float	
overUnderOpen - Float	
spread - Float	
spreadOpen - Float	

```
{  
  "gameId": 987.65,  
  "linesProviderId": 123.45,  
  "moneylineAway": 123.45,  
  "moneylineHome": 123.45,  
  "overUnder": 987.65,  
  "overUnderOpen": 123.45,  
  "spread": 123.45,  
  "spreadOpen": 987.65  
}
```

Types

GameLinesStddevPopOrderBy

order by stddevPop() on columns of
table "game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

EXAMPLE

```
{  
  "gameId": "ASC",  
  "linesProviderId": "ASC",  
  "moneylineAway": "ASC",  
  "moneylineHome": "ASC",  
  "overUnder": "ASC",  
  "overUnderOpen": "ASC",  
  "spread": "ASC",  
  "spreadOpen": "ASC"  
}
```

Types

GameLinesStddevSampFields

aggregate stddevSamp on columns

Field Name	Description
gameId - Float	
linesProviderId - Float	
moneylineAway - Float	
moneylineHome - Float	
overUnder - Float	
overUnderOpen - Float	
spread - Float	
spreadOpen - Float	

EXAMPLE

```
{  
  "gameId": 123.45,  
  "linesProviderId": 987.65,  
  "moneylineAway": 123.45,  
  "moneylineHome": 987.65,  
  "overUnder": 987.65,  
  "overUnderOpen": 123.45,  
  "spread": 987.65,  
  "spreadOpen": 123.45  
}
```

Types

GameLinesStddevSampOrderBy

order by stddevSamp() on columns of
table "game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

EXAMPLE

```
{  
  "gameId": "ASC",  
  "linesProviderId": "ASC",  
  "moneylineAway": "ASC",  
  "moneylineHome": "ASC",  
  "overUnder": "ASC",  
  "overUnderOpen": "ASC",  
  "spread": "ASC",  
  "spreadOpen": "ASC"  
}
```

Types

GameLinesSumFields

aggregate sum on columns

Field Name	Description
gameId - Int	
linesProviderId - Int	
moneylineAway - Int	
moneylineHome - Int	
overUnder - numeric	
overUnderOpen - numeric	
spread - numeric	

EXAMPLE

```
{  
  "gameId": 987,  
  "linesProviderId": 123,  
  "moneylineAway": 123,  
  "moneylineHome": 987,  
  "overUnder": numeric,  
  "overUnderOpen": numeric,  
  "spread": numeric,  
  "spreadOpen": numeric  
}
```

Field Name	Description
spreadOpen - numeric	

Types

GameLinesSumOrderBy

order by sum() on columns of table
 "game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
  "spreadOpen": "ASC"
}
```

Input Field	Description
overUnder0open -	
OrderBy	
spread - OrderBy	
spread0open -	
OrderBy	

Types

GameLinesVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
gameId - Float	
linesProviderId -	
Float	
moneylineAway -	
Float	

```
{
  "gameId": 123.45,
  "linesProviderId": 123.45,
  "moneylineAway": 987.65,
  "moneylineHome": 123.45,
  "overUnder": 123.45,
  "overUnderOpen": 123.45,
  "spread": 987.65,
  "spreadOpen": 123.45
}
```

Field Name	Description
moneylineHome -	
Float	
overUnder -	Float
overUnderOpen -	
Float	
spread -	Float
spreadOpen -	Float

Types

GameLinesVarPopOrderBy

order by varPop() on columns of table
 "game_lines"

Input Field	Description
gameId -	OrderBy
linesProviderId -	
OrderBy	

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
```

Input Field	Description	
moneylineAway -		"spreadOpen": "ASC"
OrderBy		}
moneylineHome -		
OrderBy		
overUnder -		
OrderBy		
overUnderOpen -		
OrderBy		
spread - OrderBy		
spreadOpen -		
OrderBy		

Types

GameLinesVarSampFields

aggregate varSamp on columns

EXAMPLE

```
{
  "gameId": 123.45,
  "linesProviderId": 123.45,
```

Field Name	Description	
gameId - Float		"moneylineAway": 123.45, "moneylineHome": 123.45, "overUnder": 987.65, "overUnderOpen": 123.45, "spread": 123.45, "spreadOpen": 123.45
linesProviderId - Float		}
moneylineAway - Float		
moneylineHome - Float		
overUnder - Float		
overUnderOpen - Float		
spread - Float		
spreadOpen - Float		

Types

GameLinesVarSampOrderBy

order by varSamp() on columns of table
"game_lines"

EXAMPLE

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

```
{  
  "gameId": "ASC",  
  "linesProviderId": "ASC",  
  "moneylineAway": "ASC",  
  "moneylineHome": "ASC",  
  "overUnder": "ASC",  
  "overUnderOpen": "ASC",  
  "spread": "ASC",  
  "spreadOpen": "ASC"  
}
```

Types

GameLinesVarianceFields

aggregate variance on columns

Field Name	Description
gameId - Float	
linesProviderId - Float	
moneylineAway - Float	
moneylineHome - Float	
overUnder - Float	
overUnderOpen - Float	
spread - Float	
spreadOpen - Float	

EXAMPLE

```
{  
  "gameId": 123.45,  
  "linesProviderId": 987.65,  
  "moneylineAway": 123.45,  
  "moneylineHome": 987.65,  
  "overUnder": 123.45,  
  "overUnderOpen": 987.65,  
  "spread": 987.65,  
  "spreadOpen": 987.65  
}
```

Types

GameLinesVarianceOrderBy

order by variance() on columns of table
"game_lines"

Input Field	Description
gameId - OrderBy	
linesProviderId - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
overUnder - OrderBy	
overUnderOpen - OrderBy	
spread - OrderBy	
spreadOpen - OrderBy	

EXAMPLE

```
{
  "gameId": "ASC",
  "linesProviderId": "ASC",
  "moneylineAway": "ASC",
  "moneylineHome": "ASC",
  "overUnder": "ASC",
  "overUnderOpen": "ASC",
  "spread": "ASC",
  "spreadOpen": "ASC"
}
```

Types

GameMedia

columns and relationships of
"game_media"

Field Name	Description
mediaType - media_type!	
name - String!	

EXAMPLE

```
{  
  "mediaType": media_type,  
  "name": "xyz789"  
}
```

Types

GameMediaAggregateOrderBy

order by aggregate values of table
"game_media"

Input Field	Description
count - OrderBy	

EXAMPLE

```
{  
  "count": "ASC",  
  "max": GameMediaMaxOrderBy,  
  "min": GameMediaMinOrderBy  
}
```

Input Field	Description
max -	GameMediaMaxOrderBy
min -	GameMediaMinOrderBy

Types

GameMediaBoolExp

Boolean expression to filter rows from the table "game_media". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[GameMediaBoolExp!]
_not -	GameMediaBoolExp
_or -	[GameMediaBoolExp!]

EXAMPLE

```
{  
  "_and": [GameMediaBoolExp],  
  "_not": GameMediaBoolExp,  
  "_or": [GameMediaBoolExp],  
  "mediaType": MediaTypeComparisonE  
  "name": StringComparisonExp  
}
```

Input Field	Description
mediaType -	MediaTypeComparisonExp
name -	StringComparisonExp

Types

GameMediaMaxOrderBy

order by max() on columns of table
"game_media"

EXAMPLE

```
{"mediaType": "ASC", "name": "ASC"}
```

Input Field	Description
mediaType -	OrderBy
name -	OrderBy

Types

GameMediaMinOrderBy

order by min() on columns of table
"game_media"

EXAMPLE

```
{"mediaType": "ASC", "name": "ASC"}
```

Input Field	Description
mediaType -	
OrderBy	
name -	OrderBy

Types

GameMediaOrderBy

Ordering options when selecting data from "game_media".

EXAMPLE

```
{"mediaType": "ASC", "name": "ASC"}
```

Input Field	Description
mediaType -	
OrderBy	

Input Field	Description
name - OrderBy	

Types

GameMediaSelectColumn

select columns of table "game_media"

EXAMPLE

Enum Value	Description	"mediaType"
mediaType	column name	
name	column name	

Types

GamePlayerStat

columns and relationships of
"game_player_stat"

Field Name	Description
athlete - Athlete!	An object relationship
athleteId - bigint!	
gameTeam - GameTeam!	An object relationship
gameTeamId - bigint!	
id - bigint!	
playerStatCategory - PlayerStatCategory!	An object relationship
playerStatType - PlayerStatType!	An object relationship
stat - String!	

EXAMPLE

```
{
  "athlete": Athlete,
  "athleteId": bigint,
  "gameTeam": GameTeam,
  "gameTeamId": bigint,
  "id": bigint,
  "playerStatCategory": PlayerStatCategory,
  "playerStatType": PlayerStatType,
  "stat": "abc123"
}
```

Types

GamePlayerStatAggregate

aggregated selection of
"game_player_stat"

Field Name	Description
aggregate -	GamePlayerStatAggregateFields
nodes -	[GamePlayerStat!]!

EXAMPLE

```
{  
  "aggregate": GamePlayerStatAggregat  
  "nodes": [GamePlayerStat]  
}
```

Types

GamePlayerStatAggregateBoolExp

Input Field	Description
count -	gamePlayerStatAggregateBoolExpCount

EXAMPLE

```
{"count": gamePlayerStatAggregateBo
```

Types

GamePlayerStatAggregateFields

aggregate fields of "game_player_stat"

Field Name	Description
avg -	GamePlayerStatAvgFields
count - Int!	<div><p>Arguments</p><p>columns - [GamePlayerStatSelectColumn!]</p><p>distinct - Boolean</p></div>
max -	GamePlayerStatMaxFields
min -	GamePlayerStatMinFields
stddev -	GamePlayerStatStddevFields

EXAMPLE

```
{  
  "avg": GamePlayerStatAvgFields,  
  "count": 987,  
  "max": GamePlayerStatMaxFields,  
  "min": GamePlayerStatMinFields,  
  "stddev": GamePlayerStatStddevFie  
  "stddevPop": GamePlayerStatStddev  
  "stddevSamp": GamePlayerStatStdde  
  "sum": GamePlayerStatSumFields,  
  "varPop": GamePlayerStatVarPopFie  
  "varSamp": GamePlayerStatVarSampF  
  "variance": GamePlayerStatVarianc  
}
```

Field Name	Description
stddevPop -	GamePlayerStatStddevPopFields
stddevSamp -	GamePlayerStatStddevSampFields
sum -	GamePlayerStatSumFields
varPop -	GamePlayerStatVarPopFields
varSamp -	GamePlayerStatVarSampFields
variance -	GamePlayerStatVarianceFields

Types

GamePlayerStatAggregateOrderBy

order by aggregate values of table
"game_player_stat"

EXAMPLE

```
{  
  "avg": GamePlayerStatAvgOrderBy,
```

Input Field	Description
avg -	GamePlayerStatAvgOrderBy
count - OrderBy	
max -	GamePlayerStatMaxOrderBy
min -	GamePlayerStatMinOrderBy
stddev -	GamePlayerStatStddevOrderBy
stddevPop -	GamePlayerStatStddevPopOrderBy
stddevSamp -	GamePlayerStatStddevSampOrderBy
sum -	GamePlayerStatSumOrderBy
varPop -	GamePlayerStatVarPopOrderBy
varSamp -	GamePlayerStatVarSampOrderBy
variance -	GamePlayerStatVarianceOrderBy

```

    "count": "ASC",
    "max": GamePlayerStatMaxOrderBy,
    "min": GamePlayerStatMinOrderBy,
    "stddev": GamePlayerStatStddevOrd
    "stddevPop": GamePlayerStatStddev
    "stddevSamp": GamePlayerStatStdde
    "sum": GamePlayerStatSumOrderBy,
    "varPop": GamePlayerStatVarPopOrd
    "varSamp": GamePlayerStatVarSamp0
    "variance": GamePlayerStatVarianc
}

```

Types

GamePlayerStatAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
athleteId - Float	
gameTeamId - Float	
id - Float	

```
{"athleteId": 987.65, "gameTeamId":
```



Types

GamePlayerStatAvgOrderBy

order by avg() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```



Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id -	OrderBy

Types

GamePlayerStatBoolExp

Boolean expression to filter rows from the table "game_player_stat". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	
[GamePlayerStatBoolExp!]	
_not -	
GamePlayerStatBoolExp	

EXAMPLE

```
{
  "_and": [GamePlayerStatBoolExp],
  "_not": GamePlayerStatBoolExp,
  "_or": [GamePlayerStatBoolExp],
  "athlete": AthleteBoolExp,
  "athleteId": BigIntComparisonExp,
  "gameTeam": GameTeamBoolExp,
  "gameTeamId": BigIntComparisonExp
  "id": BigIntComparisonExp,
  "playerStatCategory": PlayerStatC
  "playerStatType": PlayerStatTypeB
```

Input Field	Description	
_or -		"stat": StringComparisonExp }
	[GamePlayerStatBoolExp!]	
athlete -		
	AthleteBoolExp	
athleteId -		
	BigintComparisonExp	
gameTeam -		
	GameTeamBoolExp	
gameTeamId -		
	BigintComparisonExp	
id -		
	BigintComparisonExp	
playerStatCategory		
-		
	PlayerStatCategoryBoolExp	
playerStatType -		
	PlayerStatTypeBoolExp	
stat -		
	StringComparisonExp	

Types

GamePlayerStatMaxFields

aggregate max on columns

Field Name	Description
athleteId - bigrnt	
gameTeamId - bigrnt	
id - bigrnt	
stat - String	

EXAMPLE

```
{  
  "athleteId": bigint,  
  "gameTeamId": bigint,  
  "id": bigint,  
  "stat": "abc123"  
}
```

Types

GamePlayerStatMaxOrderBy

order by max() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id - OrderBy	
stat - OrderBy	

Types

GamePlayerStatMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
athleteId - bigrnt	
gameTeamId -	
bigrnt	
id - bigrnt	
stat - String	

```
{  
    "athleteId": bigint,  
    "gameTeamId": bigint,  
    "id": bigint,  
    "stat": "xyz789"  
}
```

Types

GamePlayerStatMinOrderBy

order by min() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id - OrderBy	
stat - OrderBy	



Types

GamePlayerStatOrderBy

Ordering options when selecting data from "game_player_stat".

Input Field	Description
athlete -	AthleteOrderBy
athleteId -	OrderBy
gameTeam -	GameTeamOrderBy
gameTeamId -	OrderBy
id -	OrderBy
playerStatCategory	-
	PlayerStatCategoryOrderBy
playerStatType -	PlayerStatTypeOrderBy
stat -	OrderBy

EXAMPLE

```
{  
  "athlete": AthleteOrderBy,  
  "athleteId": "ASC",  
  "gameTeam": GameTeamOrderBy,  
  "gameTeamId": "ASC",  
  "id": "ASC",  
  "playerStatCategory": PlayerStatC  
  "playerStatType": PlayerStatType0  
  "stat": "ASC"  
}
```

Types

GamePlayerStatSelectColumn

select columns of table

"game_player_stat"

EXAMPLE

"athleteId"

Enum Value	Description
athleteId	column name
gameTeamId	column name
id	column name
stat	column name

Types

GamePlayerStatStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
athleteId - Float	
gameTeamId - Float	
id - Float	

Types

GamePlayerStatStddevOrderBy

order by stddev() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId - OrderBy	
gameTeamId - OrderBy	
id - OrderBy	

Types

GamePlayerStatStddevPopFields

aggregate stddevPop on columns

EXAMPLE

```
{"athleteId": 987.65, "gameTeamId":
```

Field Name	Description
athleteId - Float	
gameTeamId - Float	
id - Float	

Types

GamePlayerStatStddevPopOrderBy

order by stddevPop() on columns of
table "game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id -	OrderBy

Types

GamePlayerStatStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
athleteId -	Float
gameTeamId -	Float
id -	Float

{"athleteId": 987.65, "gameTeamId":



Types

GamePlayerStatStddevSampOrderBy

order by stddevSamp() on columns of
table "game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id -	OrderBy

Types

GamePlayerStatSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
athleteId - bigint	
gameTeamId - bigint	
id - bigint	

```
{
  "athleteId": bigint,
  "gameTeamId": bigint,
  "id": bigint
}
```

Types

GamePlayerStatSumOrderBy

order by sum() on columns of table
 "game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

Input Field	Description
athleteId - OrderBy	
gameTeamId - OrderBy	
id - OrderBy	



Types

GamePlayerStatVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
athleteId - Float	
gameTeamId - Float	
id - Float	

```
{"athleteId": 123.45, "gameTeamId":
```



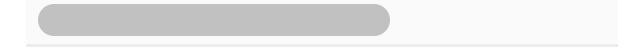
Types

GamePlayerStatVarPopOrderBy

order by varPop() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```



Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id -	OrderBy

Types

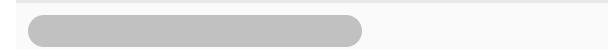
GamePlayerStatVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
athleteId -	Float
gameTeamId -	Float
id -	Float

{"athleteId": 123.45, "gameTeamId":



Types

GamePlayerStatVarSampOrderBy

order by varSamp() on columns of table
"game_player_stat"

EXAMPLE

```
{"athleteId": "ASC", "gameTeamId":
```

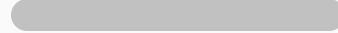
Input Field	Description
athleteId -	
OrderBy	
gameTeamId -	
OrderBy	
id -	OrderBy

Types

GamePlayerStatVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
athleteId - Float	<code>{"athleteId": 123.45, "gameTeamId":</code> 
gameTeamId - Float	
id - Float	

Types

GamePlayerStatVarianceOrderBy

order by variance() on columns of table
"game_player_stat"

EXAMPLE

```
{ "athleteId": "ASC", "gameTeamId":
```



Input Field	Description
athleteId - OrderBy	
gameTeamId - OrderBy	
id - OrderBy	

Types

GameStatusComparisonExp

Boolean expression to compare columns of type "game_status". All fields are combined with logical 'AND'.

Input Field	Description
_eq - <code>game_status</code>	
_gt - <code>game_status</code>	
_gte - <code>game_status</code>	
_in -	
<code>[game_status!]</code>	
_isNull - <code>Boolean</code>	
_lt - <code>game_status</code>	
_lte - <code>game_status</code>	
_neq - <code>game_status</code>	
_nin -	
<code>[game_status!]</code>	

EXAMPLE

```
{  
  "_eq": game_status,  
  "_gt": game_status,  
  "_gte": game_status,  
  "_in": [game_status],  
  "_isNull": false,  
  "_lt": game_status,  
  "_lte": game_status,  
  "_neq": game_status,  
  "_nin": [game_status]  
}
```

Types

GameTeam

columns and relationships of
"game_team"

Field Name	Description
endElo - Int	
game - game	An object relationship
gameId - Int!	
gamePlayerStats - [GamePlayerStat!]!	An array relationship

Arguments

distinctOn -
[\[GamePlayerStatSelectColumn!\]](#)

distinct select on columns

limit - [Int](#)

limit the number of rows returned

offset - [Int](#)

EXAMPLE

```
{  
  "endElo": 987,  
  "game": game,  
  "gameId": 987,  
  "gamePlayerStats": [GamePlayerStat],  
  "gamePlayerStatsAggregate": GamePlayerStatsAggregate,  
  "homeAway": home_away,  
  "lineScores": [smallint],  
  "points": smallint,  
  "startElo": 987,  
  "teamId": 123,  
  "winProb": numeric  
}
```

Field Name	Description
------------	-------------

skip - Int	skip the first n rows. Use only with order_by
------------	---

orderBy - [GamePlayerStatOrderBy!]	sort the rows by one or more columns
------------------------------------	--------------------------------------

where - GamePlayerStatBoolExp	filter the rows returned
-------------------------------	--------------------------

gamePlayerStatsAggregate	An aggregate
-	relationship
GamePlayerStatAggregate!	

Arguments

distinctOn -	[GamePlayerStatSelectColumn!]
--------------	-------------------------------

distinct	select on columns
----------	-------------------

limit - Int

limit	the number of rows returned
-------	-----------------------------

offset - Int

skip the first n rows. Use only with	order_by
--------------------------------------	----------

orderBy - [GamePlayerStatOrderBy!]

sort the rows by one or more columns

Field Name	Description
where - GamePlayerStatBoolExp	filter the rows returned
homeAway - home_away!	
lineScores - [smallint!]	
points - smallint	
startElo - Int	
teamId - Int!	
winProb - numeric	

Types

GameTeamBoolExp

Boolean expression to filter rows from the table "game_team". All fields are combined with a logical 'AND'.

EXAMPLE

```
{  
  "_and": [GameTeamBoolExp],  
  "_not": GameTeamBoolExp,
```

Input Field	Description
_and -	
[GameTeamBoolExp!]	
_not -	
GameTeamBoolExp	
_or -	
[GameTeamBoolExp!]	
endElo -	
IntComparisonExp	
game - gameBoolExp	
gameId -	
IntComparisonExp	
gamePlayerStats -	
GamePlayerStatBoolExp	
gamePlayerStatsAggregate	
-	
GamePlayerStatAggregateBoolExp	
homeAway -	
HomeAwayComparisonExp	
lineScores -	
SmallintArrayComparisonExp	
points -	
SmallintComparisonExp	

```

    "_or": [GameTeamBoolExp],
    "endElo": IntComparisonExp,
    "game": gameBoolExp,
    "gameId": IntComparisonExp,
    "gamePlayerStats": GamePlayerStat
    "gamePlayerStatsAggregate": GameP
    "homeAway": HomeAwayComparisonExp
    "lineScores": SmallintArrayCompar
    "points": SmallintComparisonExp,
    "startElo": IntComparisonExp,
    "teamId": IntComparisonExp,
    "winProb": NumericComparisonExp
}
```

Input Field	Description
startElo - IntComparisonExp	
teamId - IntComparisonExp	
winProb - NumericComparisonExp	

Types

GameTeamOrderBy

Ordering options when selecting data from "game_team".

EXAMPLE

```
{
  "endElo": "ASC",
  "game": gameOrderBy,
  "gameId": "ASC",
  "gamePlayerStatsAggregate": GameP
  "homeAway": "ASC",
  "lineScores": "ASC",
  "points": "ASC",
  "startElo": "ASC",
  "teamId": "ASC",
```

Input Field	Description
endElo - OrderBy	
game - gameOrderBy	
gameId - OrderBy	

Input Field	Description	
gamePlayerStatsAggregate		"winProb": "ASC"
-		}
GamePlayerStatAggregateOrderBy		
homeAway - OrderBy		
lineScores -		
OrderBy		
points - OrderBy		
startElo - OrderBy		
teamId - OrderBy		
winProb - OrderBy		

Types

GameTeamSelectColumn

select columns of table "game_team"

EXAMPLE

Enum Value	Description	
endElo	column name	"endElo"

Enum Value	Description
gameId	column name
homeAway	column name
lineScores	column name
points	column name
startElo	column name
teamId	column name
winProb	column name

Types

GameWeather

columns and relationships of
"game_weather"

Field Name	Description
condition - WeatherCondition	An object relationship

EXAMPLE

```
{
  "condition": WeatherCondition,
  "dewpoint": numeric,
  "gameId": 123,
  "humidity": numeric,
  "precipitation": numeric,
  "pressure": numeric,
  "snowfall": numeric,
```

Field Name	Description
dewpoint - <code>numeric</code>	
gameId - <code>Int!</code>	
humidity - <code>numeric</code>	
precipitation - <code>numeric</code>	
pressure - <code>numeric</code>	
snowfall - <code>numeric</code>	
temperature - <code>numeric</code>	
weatherConditionCode - <code>smallint</code>	
windDirection - <code>numeric</code>	
windGust - <code>numeric</code>	
windSpeed - <code>numeric</code>	

```
"temperature": numeric,  
"weatherConditionCode": smallint,  
"windDirection": numeric,  
"windGust": numeric,  
"windSpeed": numeric  
}
```

Types

GameWeatherBoolExp

Boolean expression to filter rows from the table "game_weather". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[GameWeatherBoolExp!]
_not -	GameWeatherBoolExp
_or -	[GameWeatherBoolExp!]
condition -	WeatherConditionBoolExp
dewpoint -	NumericComparisonExp
gameId -	IntComparisonExp
humidity -	NumericComparisonExp
precipitation -	NumericComparisonExp

EXAMPLE

```
{  
  "_and": [GameWeatherBoolExp],  
  "_not": GameWeatherBoolExp,  
  "_or": [GameWeatherBoolExp],  
  "condition": WeatherConditionBool  
  "dewpoint": NumericComparisonExp,  
  "gameId": IntComparisonExp,  
  "humidity": NumericComparisonExp,  
  "precipitation": NumericCompariso  
  "pressure": NumericComparisonExp,  
  "snowfall": NumericComparisonExp,  
  "temperature": NumericComparisonE  
  "weatherConditionCode": SmallintC  
  "windDirection": NumericCompariso  
  "windGust": NumericComparisonExp,  
  "windSpeed": NumericComparisonExp  
}
```

Input Field	Description
pressure -	
	NumericComparisonExp
snowfall -	
	NumericComparisonExp
temperature -	
	NumericComparisonExp
weatherConditionCode	
-	
	SmallintComparisonExp
windDirection -	
	NumericComparisonExp
windGust -	
	NumericComparisonExp
windSpeed -	
	NumericComparisonExp

Types

GameWeatherOrderBy

Ordering options when selecting data from "game_weather".

Input Field	Description
condition - WeatherConditionOrderBy	
dewpoint - OrderBy	
gameId - OrderBy	
humidity - OrderBy	
precipitation - OrderBy	
pressure - OrderBy	
snowfall - OrderBy	
temperature - OrderBy	
weatherConditionCode - OrderBy	
windDirection - OrderBy	
windGust - OrderBy	
windSpeed - OrderBy	

EXAMPLE

```
{
  "condition": WeatherConditionOrde
  "dewpoint": "ASC",
  "gameId": "ASC",
  "humidity": "ASC",
  "precipitation": "ASC",
  "pressure": "ASC",
  "snowfall": "ASC",
  "temperature": "ASC",
  "weatherConditionCode": "ASC",
  "windDirection": "ASC",
  "windGust": "ASC",
  "windSpeed": "ASC"
}
```

Types

GameWeatherSelectColumn

select columns of table "game_weather"

EXAMPLE

"dewpoint"

Enum Value	Description
dewpoint	column name
gameId	column name
humidity	column name
precipitation	column name
pressure	column name
snowfall	column name
temperature	column name
weatherConditionCode	column name
windDirection	column name
windGust	column name
windSpeed	column name

Types

HomeAwayComparisonExp

Boolean expression to compare columns of type "home_away". All fields are combined with logical 'AND'.

Input Field	Description
_eq - <code>home_away</code>	
_gt - <code>home_away</code>	
_gte - <code>home_away</code>	
_in - <code>[home_away!]</code>	
_isNull - <code>Boolean</code>	
_lt - <code>home_away</code>	
_lte - <code>home_away</code>	
_neq - <code>home_away</code>	
_nin - <code>[home_away!]</code>	

EXAMPLE

```
{  
  "_eq": home_away,  
  "_gt": home_away,  
  "_gte": home_away,  
  "_in": [home_away],  
  "_isNull": false,  
  "_lt": home_away,  
  "_lte": home_away,  
  "_neq": home_away,  
  "_nin": [home_away]  
}
```

Types

Hometown

columns and relationships of "hometown"

Field Name	Description
athletes - [Athlete!]!	An array relationship

Arguments

distinctOn - [AthleteSelectColumn!]

distinct select on columns

limit - Int

limit the number of rows returned

offset - Int

skip the first n rows. Use only with order_by

orderBy - [AthleteOrderBy!]

sort the rows by one or more columns

where - AthleteBoolExp

EXAMPLE

```
{  
  "athletes": [Athlete],  
  "athletesAggregate": AthleteAggre  
  "city": "abc123",  
  "country": "xyz789",  
  "countyFips": "xyz789",  
  "latitude": numeric,  
  "longitude": numeric,  
  "recruits": [Recruit],  
  "recruitsAggregate": RecruitAggre  
  "state": "abc123"  
}
```

Field Name	Description
------------	-------------

filter the rows returned	
--------------------------	--

athletesAggregate	An aggregate relationship
-	
AthleteAggregate!	

Arguments	
-----------	--

distinctOn - [AthleteSelectColumn!]	
---	--

distinct select on columns	
----------------------------	--

limit - Int	
-----------------------------	--

limit the number of rows returned	
-----------------------------------	--

offset - Int	
------------------------------	--

skip the first n rows. Use only with order_by	
---	--

orderBy - [AthleteOrderBy!]	
---	--

sort the rows by one or more columns	
--------------------------------------	--

where - AthleteBoolExp	
--	--

filter the rows returned	
--------------------------	--

city - String	
--------------------------------------	--

country - String	
---	--

Field Name	Description
countyFips -	
String	
latitude -	numeric
longitude -	numeric
recruits -	An array
[Recruit!]!	relationship
Arguments	
distinctOn -	[RecruitSelectColumn!]
distinct select on columns	
limit -	Int
limit the number of rows returned	
offset -	Int
skip the first n rows. Use only with	
order_by	
orderBy -	[RecruitOrderBy!]
sort the rows by one or more columns	
where -	RecruitBoolExp
filter the rows returned	

Field Name	Description
------------	-------------

recruitsAggregate	An aggregate relationship
-	RecruitAggregate!

Arguments

distinctOn - [\[RecruitSelectColumn!\]](#)

distinct select on columns

limit - [Int](#)

limit the number of rows returned

offset - [Int](#)

skip the first n rows. Use only with order_by

orderBy - [\[RecruitOrderBy!\]](#)

sort the rows by one or more columns

where - [RecruitBoolExp](#)

filter the rows returned

state - [String](#)

Types

HometownAggregate

aggregated selection of "hometown"

EXAMPLE

Field Name	Description
aggregate -	HometownAggregateFields
nodes -	[Hometown!]!

```
{  
  "aggregate": HometownAggregateFie  
  "nodes": [Hometown]  
}
```

Types

HometownAggregateFields

aggregate fields of "hometown"

EXAMPLE

Field Name	Description
avg -	HometownAvgFields
count - Int!	

```
{  
  "avg": HometownAvgFields,  
  "count": 987,  
  "max": HometownMaxFields,  
  "min": HometownMinFields,  
  "stddev": HometownStddevFields,  
  "stddevPop": HometownStddevPopFie  
  "stddevSamp": HometownStddevSampF
```

Field Name	Description
Arguments	
columns - HometownSelectColumn!	
distinct - Boolean	
max -	HometownMaxFields
min -	HometownMinFields
stddev -	HometownStddevFields
stddevPop -	HometownStddevPopFields
stddevSamp -	HometownStddevSampFields
sum -	HometownSumFields
varPop -	HometownVarPopFields
varSamp -	HometownVarSampFields
variance -	HometownVarianceFields

```
"sum": HometownSumFields,  
"varPop": HometownVarPopFields,  
"varSamp": HometownVarSampFields,  
"variance": HometownVarianceField  
}
```

Types

HometownAvgFields

aggregate avg on columns

EXAMPLE

```
{"latitude": 987.65, "longitude": 9
```

Field Name	Description
latitude - Float	
longitude - Float	

Types

HometownBoolExp

Boolean expression to filter rows from the table "hometown". All fields are combined with a logical 'AND'.

EXAMPLE

```
{
  "_and": [HometownBoolExp],
  "_not": HometownBoolExp,
  "_or": [HometownBoolExp],
  "athletes": AthleteBoolExp,
  "athletesAggregate": AthleteAggre
  "city": StringComparisonExp,
```

Input Field	Description	
_and - [HometownBoolExp!]		"country": StringComparisonExp, "countyFips": StringComparisonExp "latitude": NumericComparisonExp, "longitude": NumericComparisonExp "recruits": RecruitBoolExp, "recruitsAggregate": RecruitAggre
_not - HometownBoolExp		"state": StringComparisonExp }
_or - [HometownBoolExp!]		
athletes - AthleteBoolExp		
athletesAggregate -		AthleteAggregateBoolExp
city - StringComparisonExp		
country - StringComparisonExp		
countyFips - StringComparisonExp		
latitude - NumericComparisonExp		
longitude - NumericComparisonExp		

Input Field	Description
recruits -	
	RecruitBoolExp
recruitsAggregate	
-	
	RecruitAggregateBoolExp
state -	
	StringComparisonExp

Types

HometownMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
city - String	
country - String	
countyFips -	
String	

```
{
  "city": "xyz789",
  "country": "abc123",
  "countyFips": "xyz789",
  "latitude": numeric,
  "longitude": numeric,
  "state": "xyz789"
}
```

Field Name	Description
latitude - numeric	
longitude - numeric	
state - String	

Types

HometownMinFields

aggregate min on columns

EXAMPLE

```
{  
  "city": "abc123",  
  "country": "abc123",  
  "countyFips": "abc123",  
  "latitude": numeric,  
  "longitude": numeric,  
  "state": "xyz789"  
}
```

Field Name	Description
city - String	
country - String	
countyFips - String	
latitude - numeric	
longitude - numeric	

Field Name	Description
state - String	

Types

HometownOrderBy

Ordering options when selecting data from "hometown".

Input Field	Description
athletesAggregate	
-	
AthleteAggregateOrderBy	
city - OrderBy	
country - OrderBy	
countyFips -	
OrderBy	
latitude - OrderBy	

EXAMPLE

```
{  
  "athletesAggregate": AthleteAggre  
  "city": "ASC",  
  "country": "ASC",  
  "countyFips": "ASC",  
  "latitude": "ASC",  
  "longitude": "ASC",  
  "recruitsAggregate": RecruitAggre  
  "state": "ASC"  
}
```

Input Field	Description
longitude -	
OrderBy	
recruitsAggregate	
-	
RecruitAggregateOrderBy	
state - OrderBy	

Types

HometownSelectColumn

select columns of table "hometown"

[EXAMPLE](#)

Enum Value	Description	"city"
city	column name	
country	column name	
countyFips	column name	
latitude	column name	
longitude	column name	

Enum Value	Description
state	column name

Types

HometownStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

{"latitude": 987.65, "longitude": 1



Types

HometownStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

{ "latitude": 987.65, "longitude": 9 }

Types

HometownStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

{ "latitude": 987.65, "longitude": 1 }

Types

HometownSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
latitude - numeric	
longitude - numeric	

```
{  
  "latitude": numeric,  
  "longitude": numeric  
}
```

Types

HometownVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

```
{"latitude": 123.45, "longitude": 1
```

Types

HometownVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

```
{"latitude": 123.45, "longitude": 9
```

Types

HometownVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
latitude - Float	
longitude - Float	

```
{"latitude": 123.45, "longitude": 1
```

Types

Int

EXAMPLE

987

Types

IntComparisonExp

Boolean expression to compare columns of type "Int". All fields are combined with logical 'AND'.

Input Field	Description
_eq - Int	
_gt - Int	

EXAMPLE

```
{  
  "_eq": 987,  
  "_gt": 987,  
  "_gte": 123,  
  "_in": [123],  
  "_isNull": true,  
  "_lt": 987,  
  "_lte": 123,  
  "_neq": 123,
```

Input Field	Description	
_gte - Int		"_nin": [987] }
_in - [Int!]		
_isNull - Boolean		
_lt - Int		
_lte - Int		
_neq - Int		
_nin - [Int!]		

Types

LinesProvider

columns and relationships of
"lines_provider"

EXAMPLE

```
{"id": 123, "name": "xyz789"}
```

Field Name	Description
id - Int!	
name - String!	

Types

LinesProviderAggregate

aggregated selection of "lines_provider"

EXAMPLE

```
{  
  "aggregate": LinesProviderAggregate  
  "nodes": [LinesProvider]  
}
```

Field Name	Description
aggregate -	LinesProviderAggregateFields
nodes -	[LinesProvider!]!

Types

LinesProviderAggregateFields

aggregate fields of "lines_provider"

EXAMPLE

Field Name	Description
avg -	LinesProviderAvgFields
count - Int!	<div><p>Arguments</p><hr/><p>columns - [LinesProviderSelectColumn!]</p><hr/><p>distinct - Boolean</p></div>
max -	LinesProviderMaxFields
min -	LinesProviderMinFields
stddev -	LinesProviderStddevFields
stddevPop -	LinesProviderStddevPopFields
stddevSamp -	LinesProviderStddevSampFields
sum -	LinesProviderSumFields
varPop -	LinesProviderVarPopFields

```
{  
  "avg": LinesProviderAvgFields,  
  "count": 123,  
  "max": LinesProviderMaxFields,  
  "min": LinesProviderMinFields,  
  "stddev": LinesProviderStddevFiel  
  "stddevPop": LinesProviderStddevP  
  "stddevSamp": LinesProviderStddev  
  "sum": LinesProviderSumFields,  
  "varPop": LinesProviderVarPopFiel  
  "varSamp": LinesProviderVarSampFi  
  "variance": LinesProviderVariance  
}
```

Field Name	Description
varSamp -	LinesProviderVarSampFields
variance -	LinesProviderVarianceFields

Types

LinesProviderAvgFields

aggregate avg on columns

EXAMPLE

{ "id": 123.45 }

Field Name	Description
id - Float	

Types

LinesProviderBoolExp

Boolean expression to filter rows from the table "lines_provider". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[LinesProviderBoolExp!]
_not -	LinesProviderBoolExp
_or -	[LinesProviderBoolExp!]
id -	IntComparisonExp
name -	StringComparisonExp

EXAMPLE

```
{
  "_and": [LinesProviderBoolExp],
  "_not": LinesProviderBoolExp,
  "_or": [LinesProviderBoolExp],
  "id": IntComparisonExp,
  "name": StringComparisonExp
}
```



Types

LinesProviderMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description	
id - Int		{"id": 123, "name": "xyz789"}
name - String		

Types

LinesProviderMinFields

aggregate min on columns

EXAMPLE

Field Name	Description	
id - Int		{"id": 987, "name": "xyz789"}
name - String		

Types

LinesProviderOrderBy

Ordering options when selecting data from "lines_provider".

EXAMPLE

```
{"id": "ASC", "name": "ASC"}
```

Input Field	Description
id - OrderBy	
name - OrderBy	

Types

LinesProviderSelectColumn

select columns of table "lines_provider"

EXAMPLE

Enum Value	Description	"id"
id	column name	
name	column name	

Types

LinesProviderStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

LinesProviderStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

LinesProviderStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name

Description

{"id": 987.65}

id - [Float](#)

Types

LinesProviderSumFields

aggregate sum on columns

EXAMPLE

Field Name

Description

{"id": 123}

id - [Int](#)

Types

LinesProviderVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

LinesProviderVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

LinesProviderVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
------------	-------------

`id` - [Float](#)

```
{"id": 123.45}
```

Types

MediaTypeComparisonExp

Boolean expression to compare columns
of type "media_type". All fields are
combined with logical 'AND'.

EXAMPLE

Input Field	Description
-------------	-------------

`_eq` - [media_type](#)

`_gt` - [media_type](#)

`_gte` - [media_type](#)

```
{  
  "_eq": media_type,  
  "_gt": media_type,  
  "_gte": media_type,  
  "_in": [media_type],  
  "_isNull": false,  
  "_lt": media_type,  
  "_lte": media_type,  
  "_neq": media_type,  
  "_nin": [media_type]  
}
```

Input Field	Description
_in - [media_type!]	
_isNull - Boolean	
_lt - media_type	
_lte - media_type	
_neq - media_type	
_nin - [media_type!]	

Types

NumericComparisonExp

Boolean expression to compare columns of type "numeric". All fields are combined with logical 'AND'.

EXAMPLE

```
{
  "_eq": numeric,
  "_gt": numeric,
  "_gte": numeric,
  "_in": [numeric],
  "_isNull": true,
  "_lt": numeric,
  "_lte": numeric,
```

Input Field	Description
_eq - numeric	

Input Field	Description	
_gt - numeric		"_neq": numeric, "_nin": [numeric] }
_gte - numeric		
_in - [numeric!]		
_isNull - Boolean		
_lt - numeric		
_lte - numeric		
_neq - numeric		
_nin - [numeric!]		

Types

OrderBy

column ordering options

EXAMPLE

"ASC"

Enum Value	Description
ASC	in ascending order, nulls last

Enum Value	Description
ASC_NULLS_FIRST	in ascending order, nulls first
ASC_NULLS_LAST	in ascending order, nulls last
DESC	in descending order, nulls first
DESC_NULLS_FIRST	in descending order, nulls first
DESC_NULLS_LAST	in descending order, nulls last

Types

PlayerAdjustedMetricTypeComparis...

Boolean expression to compare columns
of type "player_adjusted_metric_type".
All fields are combined with logical
'AND'.

EXAMPLE

```
{
  "_eq": player_adjusted_metric_typ
  "_gt": player_adjusted_metric_typ
  "_gte": player_adjusted_metric_ty
  "_in": [player_adjusted_metric_ty
```

Input Field	Description
_eq -	<code>player_adjusted_metric_type</code>
_gt -	<code>player_adjusted_metric_type</code>
_gte -	<code>player_adjusted_metric_type</code>
_in -	<code>[player_adjusted_metric_type!]</code>
_isNull -	<code>Boolean</code>
_lt -	<code>player_adjusted_metric_type</code>
_lte -	<code>player_adjusted_metric_type</code>
_neq -	<code>player_adjusted_metric_type</code>
_nin -	<code>[player_adjusted_metric_type!]</code>

```

    "_isNull": true,
    "_lt": player_adjusted_metric_type
    "_lte": player_adjusted_metric_type
    "_neq": player_adjusted_metric_type
    "_nin": [player_adjusted_metric_type]
}
```

Types

PlayerStatCategory

columns and relationships of
"player_stat_category"

Field Name	Description
gamePlayerStats - [GamePlayerStat!]!	An array relationship

EXAMPLE

```
{  
  "gamePlayerStats": [GamePlayerSta  
  "gamePlayerStatsAggregate": GameP  
  "name": "xyz789"  
}
```

Field Name	Description
------------	-------------

Arguments	
-----------	--

distinctOn -
[GamePlayerStatSelectColumn!]

distinct select on columns

limit - Int

limit the number of rows returned

offset - Int

skip the first n rows. Use only with
order_by

orderBy - [GamePlayerStatOrderBy!]

sort the rows by one or more columns

where - GamePlayerStatBoolExp

filter the rows returned

gamePlayerStatsAggregate
An aggregate
-
relationship
GamePlayerStatAggregate!

Arguments	
-----------	--

distinctOn -
[GamePlayerStatSelectColumn!]

Field Name	Description
------------	-------------

distinct	select on columns
----------	-------------------

limit - Int	
-----------------------------	--

limit	the number of rows returned
-------	-----------------------------

offset - Int	
------------------------------	--

skip	the first n rows. Use only with order_by
------	---

orderBy - [GamePlayerStatOrderBy!]	
--	--

sort	the rows by one or more columns
------	---------------------------------

where - GamePlayerStatBoolExp	
---	--

filter	the rows returned
--------	-------------------

name - String!	
--------------------------------	--

Types

PlayerStatCategoryAggregate

aggregated selection of
"player_stat_category"

EXAMPLE

Field Name	Description
aggregate -	
PlayerStatCategoryAggregateFields	
nodes -	
[PlayerStatCategory!]!	

```
{  
    "aggregate": PlayerStatCategoryAg  
    "nodes": [PlayerStatCategory]  
}
```

Types

PlayerStatCategoryAggregateFields

aggregate fields of
"player_stat_category"

Field Name	Description
count - Int!	

EXAMPLE

```
{  
    "count": 987,  
    "max": PlayerStatCategoryMaxField  
    "min": PlayerStatCategoryMinField  
}
```

Arguments

columns -
[PlayerStatCategorySelectColumn!]

distinct - Boolean

Field Name	Description
max -	PlayerStatCategoryMaxFields
min -	PlayerStatCategoryMinFields

Types

PlayerStatCategoryBoolExp

Boolean expression to filter rows from the table "player_stat_category". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[PlayerStatCategoryBoolExp!]
_not -	PlayerStatCategoryBoolExp
_or -	[PlayerStatCategoryBoolExp!]

EXAMPLE

```
{
  "_and": [PlayerStatCategoryBoolExp
  "_not": PlayerStatCategoryBoolExp
  "_or": [PlayerStatCategoryBoolExp
  "gamePlayerStats": GamePlayerStat
  "gamePlayerStatsAggregate": GameP
  "name": StringComparisonExp
}
```

Input Field	Description
gamePlayerStats -	
	GamePlayerStatBoolExp
gamePlayerStatsAggregate	-
	GamePlayerStatAggregateBoolExp
name -	
	StringComparisonExp

Types

PlayerStatCategoryMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description	
name - String		{ "name": "xyz789" }

Types

PlayerStatCategoryMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
------------	-------------

name - String	{ "name": "abc123" }
-------------------------------	----------------------

Types

PlayerStatCategoryOrderBy

Ordering options when selecting data from "player_stat_category".

EXAMPLE

Input Field	Description
-------------	-------------

gamePlayerStatsAggregate	{ "gamePlayerStatsAggregate": GameP }
--------------------------	---------------------------------------

-	}
---	---

GamePlayerStatAggregateOrderBy	
--	--

name - OrderBy	
--------------------------------	--

Types

PlayerStatCategorySelectColumn

select columns of table

"player_stat_category"

EXAMPLE

"name"

Enum Value	Description
name	column name

Types

PlayerStatType

columns and relationships of

"player_stat_type"

EXAMPLE

{"name": "xyz789"}

Field Name	Description
name - <i>String!</i>	

Types

PlayerStatTypeAggregate

aggregated selection of
"player_stat_type"

Field Name	Description
aggregate -	PlayerStatTypeAggregateFields
nodes -	[PlayerStatType!]!

EXAMPLE

```
{  
  "aggregate": PlayerStatTypeAggreg  
  "nodes": [PlayerStatType]  
}
```

Types

PlayerStatTypeAggregateFields

aggregate fields of "player_stat_type"

EXAMPLE

```
{  
  "count": 987,  
  "max": PlayerStatTypeMaxFields,  
}
```

Field Name	Description	
count - <code>Int!</code>		<code>"min": PlayerStatTypeMinFields</code> }
	Arguments	
	columns - <code>[PlayerStatTypeSelectColumn!]</code>	
	distinct - <code>Boolean</code>	
max -		
	<code>PlayerStatTypeMaxFields</code>	
min -		
	<code>PlayerStatTypeMinFields</code>	

Types

PlayerStatTypeBoolExp

Boolean expression to filter rows from the table "player_stat_type". All fields are combined with a logical 'AND'.

EXAMPLE

```
{
  "_and": [PlayerStatTypeBoolExp],
  "_not": PlayerStatTypeBoolExp,
  "_or": [PlayerStatTypeBoolExp],
```

Input Field	Description	
_and -		"name": StringComparisonExp
	[PlayerStatTypeBoolExp!]	}
_not -		
	PlayerStatTypeBoolExp	
_or -		
	[PlayerStatTypeBoolExp!]	
name -		
	StringComparisonExp	

Types

PlayerStatTypeMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description	
name - String		{"name": "xyz789"}

Types

PlayerStatTypeMinFields

aggregate min on columns

EXAMPLE

Field Name	Description	
name - <code>String</code>		<code>{"name": "abc123"}</code>

Types

PlayerStatTypeOrderBy

Ordering options when selecting data from "player_stat_type".

EXAMPLE

`{"name": "ASC"}`

Input Field	Description
name - <code>OrderBy</code>	

Types

PlayerStatTypeSelectColumn

select columns of table

"player_stat_type"

EXAMPLE

"name"

Enum Value	Description
name	column name

Types

Poll

columns and relationships of "poll"

EXAMPLE

Field Name	Description
pollType - PollType!	An object relationship

```
{  
  "pollType": PollType,  
  "rankings": [PollRank],  
  "season": 123,  
  "seasonType": season_type,  
  "week": smallint  
}
```

Field Name	Description
------------	-------------

rankings - [PollRank!]!	An array relationship
-------------------------	-----------------------

Arguments

distinctOn - [PollRankSelectColumn!]

distinct select on columns

limit - Int

limit the number of rows returned

offset - Int

skip the first n rows. Use only with
order_by

orderBy - [PollRankOrderBy!]

sort the rows by one or more columns

where - PollRankBoolExp

filter the rows returned

season - Int!

seasonType -

season_type!

week - smallint!

Types

PollAggregateOrderBy

order by aggregate values of table "poll"

Input Field	Description
avg -	PollAvgOrderBy
count - OrderBy	
max -	PollMaxOrderBy
min -	PollMinOrderBy
stddev -	PollStddevOrderBy
stddevPop -	PollStddevPopOrderBy
stddevSamp -	PollStddevSampOrderBy
sum -	PollSumOrderBy

EXAMPLE

```
{  
  "avg": PollAvgOrderBy,  
  "count": "ASC",  
  "max": PollMaxOrderBy,  
  "min": PollMinOrderBy,  
  "stddev": PollStddevOrderBy,  
  "stddevPop": PollStddevPopOrderBy  
  "stddevSamp": PollStddevSampOrderBy  
  "sum": PollSumOrderBy,  
  "varPop": PollVarPopOrderBy,  
  "varSamp": PollVarSampOrderBy,  
  "variance": PollVarianceOrderBy  
}
```

Input Field	Description
varPop -	PollVarPopOrderBy
varSamp -	PollVarSampOrderBy
variance -	PollVarianceOrderBy

Types

PollAvgOrderBy

order by avg() on columns of table "poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollBoolExp

Boolean expression to filter rows from the table "poll". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[PollBoolExp!]
_not - PollBoolExp	
_or -	[PollBoolExp!]
pollType -	PollTypeBoolExp
rankings -	PollRankBoolExp
season -	IntComparisonExp
seasonType -	SeasonTypeComparisonExp

EXAMPLE

```
{  
  "_and": [PollBoolExp],  
  "_not": PollBoolExp,  
  "_or": [PollBoolExp],  
  "pollType": PollTypeBoolExp,  
  "rankings": PollRankBoolExp,  
  "season": IntComparisonExp,  
  "seasonType": SeasonTypeComparisonExp  
  "week": SmallintComparisonExp  
}
```

Input Field	Description
-------------	-------------

week -

[SmallIntComparisonExp](#)

Types

PollMaxOrderBy

order by max() on columns of table

"poll"

EXAMPLE

{ "season": "ASC", "seasonType": "AS

Input Field	Description
-------------	-------------

season - [OrderBy](#)

seasonType -

[OrderBy](#)

week - [OrderBy](#)

Types

PollMinOrderBy

order by min() on columns of table "poll"

EXAMPLE

```
{"season": "ASC", "seasonType": "AS
```

Input Field	Description
season - OrderBy	
seasonType - OrderBy	
week - OrderBy	

Types

PollOrderBy

Ordering options when selecting data from "poll".

EXAMPLE

```
{
  "pollType": PollTypeOrderBy,
  "rankingsAggregate": PollRankAggr
  "season": "ASC",
  "seasonType": "ASC",
```

Input Field	Description
pollType - PollTypeOrderBy	

Input Field	Description	
rankingsAggregate	-	"week": "ASC"
	PollRankAggregateOrderBy	}
	season - OrderBy	
	seasonType - OrderBy	
	week - OrderBy	

Types

PollRank

columns and relationships of "poll_rank"

EXAMPLE

Field Name	Description
firstPlaceVotes - smallint	
points - Int	

```
{
  "firstPlaceVotes": smallint,
  "points": 987,
  "poll": Poll,
  "rank": smallint,
  "team": currentTeams
}
```

Field Name	Description
poll - Poll!	An object relationship
rank - smallint	
team - currentTeams	An object relationship

Types

PollRankAggregateOrderBy

order by aggregate values of table
"poll_rank"

Input Field	Description
avg - PollRankAvgOrderBy	
count - OrderBy	
max - PollRankMaxOrderBy	

EXAMPLE

```
{
  "avg": PollRankAvgOrderBy,
  "count": "ASC",
  "max": PollRankMaxOrderBy,
  "min": PollRankMinOrderBy,
  "stddev": PollRankStddevOrderBy,
  "stddevPop": PollRankStddevPopOrderBy,
  "stddevSamp": PollRankStddevSampOrderBy,
  "sum": PollRankSumOrderBy,
  "varPop": PollRankVarPopOrderBy,
  "varSamp": PollRankVarSampOrderBy}
```

Input Field	Description	
min -		"variance": PollRankVarianceOrderBy
	PollRankMinOrderBy	}
stddev -		
	PollRankStddevOrderBy	
stddevPop -		
	PollRankStddevPopOrderBy	
stddevSamp -		
	PollRankStddevSampOrderBy	
sum -		
	PollRankSumOrderBy	
varPop -		
	PollRankVarPopOrderBy	
varSamp -		
	PollRankVarSampOrderBy	
variance -		
	PollRankVarianceOrderBy	

Types

PollRankAvgOrderBy

order by avg() on columns of table
"poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points":
```

Input Field	Description
firstPlaceVotes -	
OrderBy	
points - OrderBy	
rank - OrderBy	

Types

PollRankBoolExp

Boolean expression to filter rows from the table "poll_rank". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	
[PollRankBoolExp!]	

EXAMPLE

```
{
  "_and": [PollRankBoolExp],
  "_not": PollRankBoolExp,
  "_or": [PollRankBoolExp],
  "firstPlaceVotes": SmallintCompar
  "points": IntComparisonExp,
  "poll": PollBoolExp,
  "rank": SmallintComparisonExp,
```

Input Field	Description	
_not -		"team": currentTeamsBoolExp
PollRankBoolExp		}
_or -		
[PollRankBoolExp!]		
firstPlaceVotes -		
SmallintComparisonExp		
points -		
IntComparisonExp		
poll - PollBoolExp		
rank -		
SmallintComparisonExp		
team -		
currentTeamsBoolExp		

Types

PollRankMaxOrderBy

order by max() on columns of table
"poll_rank"

EXAMPLE

Input Field	Description
firstPlaceVotes - OrderBy	<code>{"firstPlaceVotes": "ASC", "points"</code>
points - OrderBy	
rank - OrderBy	

Types

PollRankMinOrderBy

order by min() on columns of table
"poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points"
```

Input Field	Description
firstPlaceVotes - OrderBy	
points - OrderBy	
rank - OrderBy	

Types

PollRankOrderBy

Ordering options when selecting data from "poll_rank".

Input Field	Description
firstPlaceVotes - OrderBy	
points - OrderBy	
poll - PollOrderBy	
rank - OrderBy	
team - currentTeamsOrderBy	

EXAMPLE

```
{  
  "firstPlaceVotes": "ASC",  
  "points": "ASC",  
  "poll": PollOrderBy,  
  "rank": "ASC",  
  "team": currentTeamsOrderBy  
}
```

Types

PollRankSelectColumn

select columns of table "poll_rank"

EXAMPLE

Enum Value	Description	"firstPlaceVotes"
firstPlaceVotes	column name	
points	column name	
rank	column name	

Types

PollRankStddevOrderBy

order by stddev() on columns of table
"poll_rank"**EXAMPLE**

{ "firstPlaceVotes": "ASC", "points":

Input Field	Description
firstPlaceVotes - OrderBy	
points - OrderBy	
rank - OrderBy	

Types

PollRankStddevPopOrderBy

order by stddevPop() on columns of
table "poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points":
```

Input Field	Description
firstPlaceVotes - OrderBy	
points - OrderBy	
rank - OrderBy	

Types

PollRankStddevSampOrderBy

order by stddevSamp() on columns of
table "poll_rank"

EXAMPLE

Input Field	Description
firstPlaceVotes - OrderBy	<code>{"firstPlaceVotes": "ASC", "points"</code>
points - OrderBy	
rank - OrderBy	

Types

PollRankSumOrderBy

order by sum() on columns of table
"poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points"
```

Input Field	Description
firstPlaceVotes - OrderBy	<code>{"firstPlaceVotes": "ASC", "points"</code>
points - OrderBy	
rank - OrderBy	

Types

PollRankVarPopOrderBy

order by varPop() on columns of table
"poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points":
```

Input Field	Description
firstPlaceVotes - OrderBy	
points - OrderBy	
rank - OrderBy	

Types

PollRankVarSampOrderBy

order by varSamp() on columns of table
"poll_rank"

EXAMPLE

Input Field	Description
firstPlaceVotes - OrderBy	<code>{"firstPlaceVotes": "ASC", "points"</code>
points - OrderBy	
rank - OrderBy	

Types

PollRankVarianceOrderBy

order by variance() on columns of table
"poll_rank"

EXAMPLE

```
{"firstPlaceVotes": "ASC", "points"
```

Input Field	Description
firstPlaceVotes - OrderBy	<code>{"firstPlaceVotes": "ASC", "points"</code>
points - OrderBy	
rank - OrderBy	

Types

PollSelectColumn

select columns of table "poll"

EXAMPLE

Enum Value	Description	
season	column name	"season"
seasonType	column name	
week	column name	

Types

PollStddevOrderBy

order by stddev() on columns of table
"poll"

EXAMPLE

{"season": "ASC", "week": "ASC"}

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollStddevPopOrderBy

order by stddevPop() on columns of
table "poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollStddevSampOrderBy

order by stddevSamp() on columns of
table "poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollSumOrderBy

order by sum() on columns of table
"poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollType

columns and relationships of "poll_type"

Field Name	Description
abbreviation - String	
id - Int!	
name - String!	
polls - [Poll!]!	An array relationship

Arguments
distinctOn - [PollSelectColumn!]
distinct select on columns
limit - Int
limit the number of rows returned
offset - Int
skip the first n rows. Use only with order_by

EXAMPLE

```
{  
  "abbreviation": "xyz789",  
  "id": 987,  
  "name": "xyz789",  
  "polls": [Poll],  
  "shortName": "abc123"  
}
```

Field Name	Description
orderBy - [PollOrderBy!]	sort the rows by one or more columns
where - PollBoolExp	filter the rows returned
shortName - String!	

Types

PollTypeBoolExp

Boolean expression to filter rows from the table "poll_type". All fields are combined with a logical 'AND'.

Input Field	Description
_and - [PollTypeBoolExp!]	

EXAMPLE

```
{
  "_and": [PollTypeBoolExp],
  "_not": PollTypeBoolExp,
  "_or": [PollTypeBoolExp],
  "abbreviation": StringComparisonE
  "id": IntComparisonExp,
  "name": StringComparisonExp,
  "polls": PollBoolExp,
```

Input Field	Description	
_not -		"shortName": StringComparisonExp
PollTypeBoolExp		}
_or -		
[PollTypeBoolExp!]		
abbreviation -		
StringComparisonExp		
id -		
IntComparisonExp		
name -		
StringComparisonExp		
polls -		
PollBoolExp		
shortName -		
StringComparisonExp		

Types

PollTypeOrderBy

Ordering options when selecting data from "poll_type".

Input Field	Description
abbreviation -	
OrderBy	
id -	OrderBy
name -	OrderBy
pollsAggregate -	
PollAggregateOrderBy	
shortName -	
OrderBy	

EXAMPLE

```
{
  "abbreviation": "ASC",
  "id": "ASC",
  "name": "ASC",
  "pollsAggregate": PollAggregateOr
  "shortName": "ASC"
}
```

Types

PollTypeSelectColumn

select columns of table "poll_type"

EXAMPLE

"abbreviation"

Enum Value	Description
abbreviation	column name
id	column name
name	column name
shortName	column name

Types

PollVarPopOrderBy

order by varPop() on columns of table

EXAMPLE

"poll"

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollVarSampOrderBy

order by varSamp() on columns of table
"poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

PollVarianceOrderBy

order by variance() on columns of table
"poll"

EXAMPLE

```
{"season": "ASC", "week": "ASC"}
```

Input Field	Description
season - OrderBy	
week - OrderBy	

Types

Position

columns and relationships of "position"

Field Name	Description
abbreviation - String!	
athletes - [Athlete!]!	An array relationship

Arguments
distinctOn - [AthleteSelectColumn!]
distinct select on columns
limit - Int
limit the number of rows returned
offset - Int
skip the first n rows. Use only with order_by
orderBy - [AthleteOrderBy!]
sort the rows by one or more columns

EXAMPLE

```
{  
  "abbreviation": "xyz789",  
  "athletes": [Athlete],  
  "athletesAggregate": AthleteAggre  
  "displayName": "xyz789",  
  "id": smallint,  
  "name": "xyz789"  
}
```

Field Name	Description
------------	-------------

where - AthleteBoolExp	
--	--

filter the rows returned

athletesAggregate	An aggregate
-	relationship
AthleteAggregate!	

Arguments	
-----------	--

distinctOn - [AthleteSelectColumn!]	
---	--

distinct select on columns

limit - Int	
-----------------------------	--

limit the number of rows returned

offset - Int	
------------------------------	--

skip the first n rows. Use only with
order_by

orderBy - [AthleteOrderBy!]	
---	--

sort the rows by one or more columns

where - AthleteBoolExp	
--	--

filter the rows returned

displayName -	
String!	

Field Name	Description
id - <code>smallint!</code>	
name - <code>String!</code>	

Types

PositionBoolExp

Boolean expression to filter rows from the table "position". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	
<code>[PositionBoolExp!]</code>	
_not -	
<code>PositionBoolExp</code>	
_or -	
<code>[PositionBoolExp!]</code>	
abbreviation -	
<code>StringComparisonExp</code>	

EXAMPLE

```
{
  "_and": [PositionBoolExp],
  "_not": PositionBoolExp,
  "_or": [PositionBoolExp],
  "abbreviation": StringComparisonE
  "athletes": AthleteBoolExp,
  "athletesAggregate": AthleteAggre
  "displayName": StringComparisonEx
  "id": SmallintComparisonExp,
  "name": StringComparisonExp
}
```



Input Field	Description
athletes -	
	AthleteBoolExp
athletesAggregate	
-	
	AthleteAggregateBoolExp
displayName -	
	StringComparisonExp
id -	
	SmallintComparisonExp
name -	
	StringComparisonExp

Types

PositionOrderBy

Ordering options when selecting data from "position".

EXAMPLE

```
{  
    "abbreviation": "ASC",  
    "athletesAggregate": AthleteAggre  
    "displayName": "ASC",
```

Input Field	Description	
abbreviation -		"id": "ASC", "name": "ASC"
OrderBy		}
athletesAggregate		
-		
AthleteAggregateOrderBy		
displayName -		
OrderBy		
id - OrderBy		
name - OrderBy		

Types

PositionSelectColumn

select columns of table "position"

EXAMPLE

"abbreviation"

Enum Value	Description
abbreviation	column name
displayName	column name

Enum Value	Description
id	column name
name	column name

Types

Recruit

columns and relationships of "recruit"

EXAMPLE

Field Name	Description
athlete - Athlete	An object relationship
college - currentTeams	An object relationship
height - Float	
hometown - Hometown	An object relationship
id - bigint!	
name - String!	

```
{
  "athlete": Athlete,
  "college": currentTeams,
  "height": 123.45,
  "hometown": Hometown,
  "id": bigint,
  "name": "xyz789",
  "overallRank": smallint,
  "position": RecruitPosition,
  "positionRank": smallint,
  "ranking": smallint,
  "rating": 123.45,
  "recruitSchool": RecruitSchool,
  "recruitType": recruit_type,
  "stars": smallint,
  "weight": smallint,
```

Field Name	Description	
overallRank -		"year": smallint
smallint		}
position -	An object	
RecruitPosition	relationship	
positionRank -		
smallint		
ranking -	smallint	
rating -	Float!	
recruitSchool -	An object	
RecruitSchool	relationship	
recruitType -		
recruit_type!		
stars -	smallint!	
weight -	smallint	
year -	smallint!	

Types

RecruitAggregate

aggregated selection of "recruit"

Field Name	Description
aggregate -	
nodes -	

RecruitAggregateFields

[Recruit!]!

EXAMPLE

```
{  
  "aggregate": RecruitAggregateFiel  
  "nodes": [Recruit]  
}
```

Types

RecruitAggregateBoolExp

Input Field	Description
count -	

recruitAggregateBoolExpCount

EXAMPLE

```
{"count": recruitAggregateBoolExpCo
```

Types

RecruitAggregateFields

aggregate fields of "recruit"

Field Name	Description
avg -	RecruitAvgFields
count - Int!	Arguments columns - [RecruitSelectColumn!]
distinct - Boolean	
max -	RecruitMaxFields
min -	RecruitMinFields
stddev -	RecruitStddevFields
stddevPop -	RecruitStddevPopFields
stddevSamp -	RecruitStddevSampFields

EXAMPLE

```
{  
  "avg": RecruitAvgFields,  
  "count": 987,  
  "max": RecruitMaxFields,  
  "min": RecruitMinFields,  
  "stddev": RecruitStddevFields,  
  "stddevPop": RecruitStddevPopFiel  
  "stddevSamp": RecruitStddevSampFi  
  "sum": RecruitSumFields,  
  "varPop": RecruitVarPopFields,  
  "varSamp": RecruitVarSampFields,  
  "variance": RecruitVarianceFields  
}
```

Field Name	Description
sum -	
	RecruitSumFields
varPop -	
	RecruitVarPopFields
varSamp -	
	RecruitVarSampFields
variance -	
	RecruitVarianceFields

Types

RecruitAggregateOrderBy

order by aggregate values of table
"recruit"

EXAMPLE

```
{
  "avg": RecruitAvgOrderBy,
  "count": "ASC",
  "max": RecruitMaxOrderBy,
  "min": RecruitMinOrderBy,
  "stddev": RecruitStddevOrderBy,
  "stddevPop": RecruitStddevPopOrde,
  "stddevSamp": RecruitStddevSampOr
  "sum": RecruitSumOrderBy,
```

Input Field	Description
avg -	
	RecruitAvgOrderBy
count - OrderBy	

Input Field	Description	
max -	RecruitMaxOrderBy	"varPop": RecruitVarPopOrderBy, "varSamp": RecruitVarSampOrderBy, "variance": RecruitVarianceOrderBy }
min -	RecruitMinOrderBy	
stddev -	RecruitStddevOrderBy	
stddevPop -	RecruitStddevPopOrderBy	
stddevSamp -	RecruitStddevSampOrderBy	
sum -	RecruitSumOrderBy	
varPop -	RecruitVarPopOrderBy	
varSamp -	RecruitVarSampOrderBy	
variance -	RecruitVarianceOrderBy	

Types

RecruitAvgFields

aggregate avg on columns

Field Name	Description
height - Float	
id - Float	
overallRank - Float	
positionRank - Float	
ranking - Float	
rating - Float	
stars - Float	
weight - Float	
year - Float	

EXAMPLE

```
{  
  "height": 123.45,  
  "id": 987.65,  
  "overallRank": 987.65,  
  "positionRank": 123.45,  
  "ranking": 987.65,  
  "rating": 987.65,  
  "stars": 123.45,  
  "weight": 987.65,  
  "year": 123.45  
}
```

Types

RecruitAvgOrderBy

order by avg() on columns of table
"recruit"

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

EXAMPLE

```
{  
  "height": "ASC",  
  "id": "ASC",  
  "overallRank": "ASC",  
  "positionRank": "ASC",  
  "ranking": "ASC",  
  "rating": "ASC",  
  "stars": "ASC",  
  "weight": "ASC",  
  "year": "ASC"  
}
```

Types

RecruitBoolExp

Boolean expression to filter rows from the table "recruit". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[RecruitBoolExp!]
_not -	RecruitBoolExp
_or -	[RecruitBoolExp!]
athlete -	AthleteBoolExp
college -	currentTeamsBoolExp
height -	FloatComparisonExp
hometown -	HometownBoolExp
id -	BigintComparisonExp

EXAMPLE

```
{  
  "_and": [RecruitBoolExp],  
  "_not": RecruitBoolExp,  
  "_or": [RecruitBoolExp],  
  "athlete": AthleteBoolExp,  
  "college": currentTeamsBoolExp,  
  "height": FloatComparisonExp,  
  "hometown": HometownBoolExp,  
  "id": BigintComparisonExp,  
  "name": StringComparisonExp,  
  "overallRank": SmallintComparison  
  "position": RecruitPositionBoolEx  
  "positionRank": SmallintCompariso  
  "ranking": SmallintComparisonExp,  
  "rating": FloatComparisonExp,  
  "recruitSchool": RecruitSchoolBoo  
  "recruitType": RecruitTypeCompa  
  "stars": SmallintComparisonExp,  
  "weight": SmallintComparisonExp,  
  "year": SmallintComparisonExp  
}
```

Input Field	Description
name -	StringComparisonExp
overallRank -	SmallintComparisonExp
position -	RecruitPositionBoolExp
positionRank -	SmallintComparisonExp
ranking -	SmallintComparisonExp
rating -	FloatComparisonExp
recruitSchool -	RecruitSchoolBoolExp
recruitType -	RecruitTypeComparisonExp
stars -	SmallintComparisonExp
weight -	SmallintComparisonExp
year -	SmallintComparisonExp

Types

RecruitMaxFields

aggregate max on columns

Field Name	Description
height - <code>Float</code>	
id - <code>bigint</code>	
name - <code>String</code>	
overallRank - <code>smallint</code>	
positionRank - <code>smallint</code>	
ranking - <code>smallint</code>	
rating - <code>Float</code>	
recruitType - <code>recruit_type</code>	
stars - <code>smallint</code>	
weight - <code>smallint</code>	
year - <code>smallint</code>	

EXAMPLE

```
{  
  "height": 123.45,  
  "id": bigint,  
  "name": "xyz789",  
  "overallRank": smallint,  
  "positionRank": smallint,  
  "ranking": smallint,  
  "rating": 123.45,  
  "recruitType": recruit_type,  
  "stars": smallint,  
  "weight": smallint,  
  "year": smallint  
}
```

Types

RecruitMaxOrderBy

order by max() on columns of table
"recruit"

Input Field	Description
height - OrderBy	
id - OrderBy	
name - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
recruitType - OrderBy	
stars - OrderBy	

EXAMPLE

```
{  
  "height": "ASC",  
  "id": "ASC",  
  "name": "ASC",  
  "overallRank": "ASC",  
  "positionRank": "ASC",  
  "ranking": "ASC",  
  "rating": "ASC",  
  "recruitType": "ASC",  
  "stars": "ASC",  
  "weight": "ASC",  
  "year": "ASC"  
}
```

Input Field	Description
weight - OrderBy	
year - OrderBy	

Types

RecruitMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
height - Float	
id - bigint	
name - String	
overallRank - smallint	
positionRank - smallint	
ranking - smallint	

```
{
  "height": 123.45,
  "id": bigint,
  "name": "abc123",
  "overallRank": smallint,
  "positionRank": smallint,
  "ranking": smallint,
  "rating": 123.45,
  "recruitType": recruit_type,
  "stars": smallint,
  "weight": smallint,
  "year": smallint
}
```

Field Name	Description
rating - <code>Float</code>	
recruitType - <code>recruit_type</code>	
stars - <code>smallint</code>	
weight - <code>smallint</code>	
year - <code>smallint</code>	

Types

RecruitMinOrderBy

order by min() on columns of table
"recruit"

Input Field	Description
height - <code>OrderBy</code>	
id - <code>OrderBy</code>	
name - <code>OrderBy</code>	

EXAMPLE

```
{
  "height": "ASC",
  "id": "ASC",
  "name": "ASC",
  "overallRank": "ASC",
  "positionRank": "ASC",
  "ranking": "ASC",
  "rating": "ASC",
  "recruitType": "ASC",
  "stars": "ASC",
  "weight": "ASC",
```

Input Field	Description	
overallRank -		"year": "ASC"
OrderBy		}
positionRank -		
OrderBy		
ranking -	OrderBy	
rating -	OrderBy	
recruitType -		
OrderBy		
stars -	OrderBy	
weight -	OrderBy	
year -	OrderBy	

Types

RecruitOrderBy

Ordering options when selecting data from "recruit".

EXAMPLE

```
{  
  "athlete": AthleteOrderBy,
```

Input Field	Description
athlete -	
AthleteOrderBy	
college -	
currentTeamsOrderBy	
height - OrderBy	
hometown -	
HometownOrderBy	
id - OrderBy	
name - OrderBy	
overallRank -	
OrderBy	
position -	
RecruitPositionOrderBy	
positionRank -	
OrderBy	
ranking - OrderBy	
rating - OrderBy	
recruitSchool -	
RecruitSchoolOrderBy	
recruitType -	
OrderBy	

```

"college": currentTeamsOrderBy,
"height": "ASC",
"hometown": HometownOrderBy,
"id": "ASC",
"name": "ASC",
"overallRank": "ASC",
"position": RecruitPositionOrderBy,
"positionRank": "ASC",
"ranking": "ASC",
"rating": "ASC",
"recruitSchool": RecruitSchoolOrderBy,
"recruitType": "ASC",
"stars": "ASC",
"weight": "ASC",
"year": "ASC"
}

```

Input Field	Description
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

Types

RecruitPosition

columns and relationships of
"recruit_position"

Field Name	Description
id - smallint!	
position - String!	
positionGroup - String	

EXAMPLE

```
{  
  "id": smallint,  
  "position": "abc123",  
  "positionGroup": "xyz789"  
}
```

Types

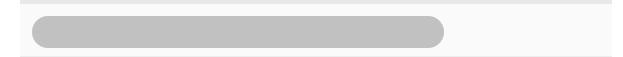
RecruitPositionAggregate

aggregated selection of
"recruit_position"

Field Name	Description
aggregate -	RecruitPositionAggregateFields
nodes -	[RecruitPosition!]!

EXAMPLE

```
{  
  "aggregate": RecruitPositionAggre  
  "nodes": [RecruitPosition]  
}
```



Types

RecruitPositionAggregateFields

aggregate fields of "recruit_position"

EXAMPLE

```
{  
  "avg": RecruitPositionAvgFields,  
  "count": 987,
```

Field Name	Description
avg -	
	RecruitPositionAvgFields
count - Int!	
	<div style="border: 1px solid #ccc; padding: 5px;"> Arguments </div>
columns -	
	[RecruitPositionSelectColumn!]?
distinct - Boolean	
max -	
	RecruitPositionMaxFields
min -	
	RecruitPositionMinFields
stddev -	
	RecruitPositionStddevFields
stddevPop -	
	RecruitPositionStddevPopFields
stddevSamp -	
	RecruitPositionStddevSampFields
sum -	
	RecruitPositionSumFields
varPop -	
	RecruitPositionVarPopFields

```

    "max": RecruitPositionMaxFields,
    "min": RecruitPositionMinFields,
    "stddev": RecruitPositionStddevFi
    "stddevPop": RecruitPositionStdde
    "stddevSamp": RecruitPositionStdd
    "sum": RecruitPositionSumFields,
    "varPop": RecruitPositionVarPopFi
    "varSamp": RecruitPositionVarSamp
    "variance": RecruitPositionVarian
  }
}

```

Field Name	Description
varSamp -	RecruitPositionVarSampFields
variance -	RecruitPositionVarianceFields

Types

RecruitPositionAvgFields

aggregate avg on columns

EXAMPLE

{ "id": 987.65 }

Field Name	Description
id - Float	

Types

RecruitPositionBoolExp

Boolean expression to filter rows from the table "recruit_position". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[RecruitPositionBoolExp!]
_not -	RecruitPositionBoolExp
_or -	[RecruitPositionBoolExp!]
id -	SmallintComparisonExp
position -	StringComparisonExp
positionGroup -	StringComparisonExp

EXAMPLE

```
{  
  "_and": [RecruitPositionBoolExp],  
  "_not": RecruitPositionBoolExp,  
  "_or": [RecruitPositionBoolExp],  
  "id": SmallintComparisonExp,  
  "position": StringComparisonExp,  
  "positionGroup": StringComparison  
}
```

Types

RecruitPositionMaxFields

aggregate max on columns

Field Name	Description
id - <code>smallint</code>	
position - <code>String</code>	
positionGroup - <code>String</code>	

EXAMPLE

```
{  
  "id": smallint,  
  "position": "abc123",  
  "positionGroup": "abc123"  
}
```

Types

RecruitPositionMinFields

aggregate min on columns

Field Name	Description
id - <code>smallint</code>	
position - <code>String</code>	
positionGroup - <code>String</code>	

EXAMPLE

```
{  
  "id": smallint,  
  "position": "abc123",  
  "positionGroup": "xyz789"  
}
```

Types

RecruitPositionOrderBy

Ordering options when selecting data from "recruit_position".

EXAMPLE

```
{"id": "ASC", "position": "ASC", "p
```

Input Field	Description
id - OrderBy	
position - OrderBy	
positionGroup - OrderBy	

Types

RecruitPositionSelectColumn

select columns of table
"recruit_position"

EXAMPLE

```
"id"
```

Enum Value	Description
id	column name
position	column name
positionGroup	column name

Types

RecruitPositionStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitPositionStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name

Description

{ "id": 987.65 }

id - [Float](#)

Types

RecruitPositionStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name

Description

{ "id": 123.45 }

id - [Float](#)

Types

RecruitPositionSumFields

aggregate sum on columns

EXAMPLE

Field Name

Description

{ "id": smallint }

id - smallint

Types

RecruitPositionVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name

Description

{ "id": 987.65 }

id - Float

Types

RecruitPositionVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitPositionVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitSchool

columns and relationships of
"recruit_school"

Field Name	Description
------------	-------------

id - <code>Int!</code>	
------------------------	--

name - <code>String!</code>	
-----------------------------	--

recruits - <code>[Recruit!]!</code>	An array relationship
--	--------------------------

Arguments

`distinctOn - [RecruitSelectColumn!]`

distinct select on columns

`limit - Int`

limit the number of rows returned

`offset - Int`

skip the first n rows. Use only with
`order_by`

`orderBy - [RecruitOrderBy!]`

sort the rows by one or more columns

`where - RecruitBoolExp`

filter the rows returned

EXAMPLE

```
{  
  "id": 123,  
  "name": "abc123",  
  "recruits": [Recruit],  
  "recruitsAggregate": RecruitAggre  
}
```

Field Name	Description
recruitsAggregate	An aggregate relationship
-	
RecruitAggregate!	

Arguments

`distinctOn - [RecruitSelectColumn!]`

distinct select on columns

`limit - Int`

limit the number of rows returned

`offset - Int`

skip the first n rows. Use only with `order_by`

`orderBy - [RecruitOrderBy!]`

sort the rows by one or more columns

`where - RecruitBoolExp`

filter the rows returned

Types

RecruitSchoolAggregate

aggregated selection of "recruit_school"

EXAMPLE

Field Name	Description
aggregate -	RecruitSchoolAggregateFields
nodes -	[RecruitSchool!]!

```
{  
  "aggregate": RecruitSchoolAggregate  
  "nodes": [RecruitSchool]  
}
```

Types

RecruitSchoolAggregateFields

aggregate fields of "recruit_school"

EXAMPLE

Field Name	Description
avg -	RecruitSchoolAvgFields
count - Int!	

```
{  
  "avg": RecruitSchoolAvgFields,  
  "count": 987,  
  "max": RecruitSchoolMaxFields,  
  "min": RecruitSchoolMinFields,  
  "stddev": RecruitSchoolStddevFiel  
  "stddevPop": RecruitSchoolStddevP  
  "stddevSamp": RecruitSchoolStddev
```

Field Name	Description
Arguments	
columns - [RecruitSchoolSelectColumn!]	
distinct - Boolean	
max -	RecruitSchoolMaxFields
min -	RecruitSchoolMinFields
stddev -	RecruitSchoolStddevFields
stddevPop -	RecruitSchoolStddevPopFields
stddevSamp -	RecruitSchoolStddevSampFields
sum -	RecruitSchoolSumFields
varPop -	RecruitSchoolVarPopFields
varSamp -	RecruitSchoolVarSampFields
variance -	RecruitSchoolVarianceFields

```
"sum": RecruitSchoolSumFields,  
"varPop": RecruitSchoolVarPopFiel  
"varSamp": RecruitSchoolVarSampFi  
"variance": RecruitSchoolVariance  
}
```

Types

RecruitSchoolAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 987.65}

Types

RecruitSchoolBoolExp

Boolean expression to filter rows from the table "recruit_school". All fields are combined with a logical 'AND'.

EXAMPLE

Input Field	Description
_and - [RecruitSchoolBoolExp!]	

```
{  
  "_and": [RecruitSchoolBoolExp],  
  "_not": RecruitSchoolBoolExp,  
  "_or": [RecruitSchoolBoolExp],  
  "id": IntComparisonExp,  
  "name": StringComparisonExp,  
  "recruits": RecruitBoolExp,
```

Input Field	Description	
_not -		"recruitsAggregate": RecruitAggre
RecruitSchoolBoolExp		}
_or -		
[RecruitSchoolBoolExp!]		
id -		
IntComparisonExp		
name -		
StringComparisonExp		
recruits -		
RecruitBoolExp		
recruitsAggregate		
-		
RecruitAggregateBoolExp		

Types

RecruitSchoolMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description	
id - Int		{"id": 123, "name": "xyz789"}
name - String		

Types

RecruitSchoolMinFields

aggregate min on columns

EXAMPLE

Field Name	Description	
id - Int		{"id": 123, "name": "xyz789"}
name - String		

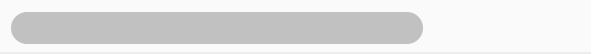
Types

RecruitSchoolOrderBy

Ordering options when selecting data from "recruit_school".

EXAMPLE

```
{
  "id": "ASC",
  "name": "ASC",
  "recruitsAggregate": RecruitAggre
}
```



Input Field	Description
id - OrderBy	
name - OrderBy	
recruitsAggregate	
-	
RecruitAggregateOrderBy	

Types

RecruitSchoolSelectColumn

select columns of table "recruit_school"

EXAMPLE

Enum Value	Description
id	column name
name	column name

Types

RecruitSchoolStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 987.65}

Types

RecruitSchoolStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 987.65}

Types

RecruitSchoolStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 987.65}

Types

RecruitSchoolSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description	
id - Int		{"id": 987}

Types

RecruitSchoolVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitSchoolVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitSchoolVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description	
id - Float		{"id": 123.45}

Types

RecruitSelectColumn

select columns of table "recruit"

EXAMPLE

Enum Value	Description	
height	column name	"height"
id	column name	
name	column name	

Enum Value	Description
overallRank	column name
positionRank	column name
ranking	column name
rating	column name
recruitType	column name
stars	column name
weight	column name
year	column name

Types

RecruitStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
height - Float	
id - Float	

```
{
  "height": 987.65,
  "id": 987.65,
  "overallRank": 123.45,
  "positionRank": 987.65,
  "ranking": 123.45,
```

Field Name	Description	
overallRank -		"rating": 987.65, "stars": 123.45, "weight": 123.45, "year": 987.65
Float		}
positionRank -		
Float		
ranking -	Float	
rating -	Float	
stars -	Float	
weight -	Float	
year -	Float	

Types

RecruitStddevOrderBy

order by stddev() on columns of table
"recruit"

EXAMPLE

```
{
  "height": "ASC",
  "id": "ASC",
  "overallRank": "ASC",
  "positionRank": "ASC",
  "ranking": "ASC",
```

Input Field	Description
height -	OrderBy

Input Field	Description	
id - OrderBy		"rating": "ASC", "stars": "ASC", "weight": "ASC", "year": "ASC"
overallRank - OrderBy		}
positionRank - OrderBy		
ranking - OrderBy		
rating - OrderBy		
stars - OrderBy		
weight - OrderBy		
year - OrderBy		

Types

RecruitStddevPopFields

aggregate stddevPop on columns

EXAMPLE

```
{
  "height": 123.45,
  "id": 123.45,
  "overallRank": 987.65,
```

Field Name	Description	
height - <code>Float</code>		"positionRank": 123.45, "ranking": 123.45, "rating": 987.65, "stars": 987.65, "weight": 987.65, "year": 123.45
id - <code>Float</code>		}
overallRank - <code>Float</code>		
positionRank - <code>Float</code>		
ranking - <code>Float</code>		
rating - <code>Float</code>		
stars - <code>Float</code>		
weight - <code>Float</code>		
year - <code>Float</code>		

Types

RecruitStddevPopOrderBy

order by stddevPop() on columns of
table "recruit"

EXAMPLE

```
{
  "height": "ASC",
```

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

```

    "id": "ASC",
    "overallRank": "ASC",
    "positionRank": "ASC",
    "ranking": "ASC",
    "rating": "ASC",
    "stars": "ASC",
    "weight": "ASC",
    "year": "ASC"
}
```

Types

RecruitStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

```
{
  "height": 123.45,
```

Field Name	Description
height - <code>Float</code>	
id - <code>Float</code>	
overallRank - <code>Float</code>	
positionRank - <code>Float</code>	
ranking - <code>Float</code>	
rating - <code>Float</code>	
stars - <code>Float</code>	
weight - <code>Float</code>	
year - <code>Float</code>	

```

    "id": 987.65,
    "overallRank": 123.45,
    "positionRank": 123.45,
    "ranking": 123.45,
    "rating": 123.45,
    "stars": 987.65,
    "weight": 123.45,
    "year": 123.45
}
```

Types

RecruitStddevSampOrderBy

order by stddevSamp() on columns of
table "recruit"

EXAMPLE

```
{
  "height": "ASC",
```

Input Field	Description
height - <code>OrderBy</code>	
id - <code>OrderBy</code>	
overallRank - <code>OrderBy</code>	
positionRank - <code>OrderBy</code>	
ranking - <code>OrderBy</code>	
rating - <code>OrderBy</code>	
stars - <code>OrderBy</code>	
weight - <code>OrderBy</code>	
year - <code>OrderBy</code>	

```
{
  "id": "ASC",
  "overallRank": "ASC",
  "positionRank": "ASC",
  "ranking": "ASC",
  "rating": "ASC",
  "stars": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Types

RecruitSumFields

aggregate sum on columns

EXAMPLE

```
{
  "height": 123.45,
```

Field Name	Description
height - <code>Float</code>	
id - <code>bigint</code>	
overallRank - <code>smallint</code>	
positionRank - <code>smallint</code>	
ranking - <code>smallint</code>	
rating - <code>Float</code>	
stars - <code>smallint</code>	
weight - <code>smallint</code>	
year - <code>smallint</code>	

```

"id": bigint,
"overallRank": smallint,
"positionRank": smallint,
"ranking": smallint,
"rating": 987.65,
"stars": smallint,
"weight": smallint,
"year": smallint
}
}
```

Types

RecruitSumOrderBy

order by sum() on columns of table
"recruit"

EXAMPLE

```
{
  "height": "ASC",
```

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

```

    "id": "ASC",
    "overallRank": "ASC",
    "positionRank": "ASC",
    "ranking": "ASC",
    "rating": "ASC",
    "stars": "ASC",
    "weight": "ASC",
    "year": "ASC"
}
```

Types

RecruitTypeComparisonExp

Boolean expression to compare columns of type "recruit_type". All fields are combined with logical 'AND'.

EXAMPLE

```
{
  "_eq": recruit_type,
```

Input Field	Description
_eq - <code>recruit_type</code>	
_gt - <code>recruit_type</code>	
_gte - <code>recruit_type</code>	
_in - [<code>recruit_type!</code>]	
_isNull - <code>Boolean</code>	
_lt - <code>recruit_type</code>	
_lte - <code>recruit_type</code>	
_neq - <code>recruit_type</code>	
_nin - [<code>recruit_type!</code>]	

```

    "_gt": recruit_type,
    "_gte": recruit_type,
    "_in": [recruit_type],
    "_isNull": false,
    "_lt": recruit_type,
    "_lte": recruit_type,
    "_neq": recruit_type,
    "_nin": [recruit_type]
}
```

Types

RecruitVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
height - Float	
id - Float	
overallRank - Float	
positionRank - Float	
ranking - Float	
rating - Float	
stars - Float	
weight - Float	
year - Float	

```
{  
  "height": 987.65,  
  "id": 987.65,  
  "overallRank": 987.65,  
  "positionRank": 987.65,  
  "ranking": 987.65,  
  "rating": 123.45,  
  "stars": 123.45,  
  "weight": 123.45,  
  "year": 987.65  
}
```

Types

RecruitVarPopOrderBy

order by varPop() on columns of table
"recruit"

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

EXAMPLE

```
{
  "height": "ASC",
  "id": "ASC",
  "overallRank": "ASC",
  "positionRank": "ASC",
  "ranking": "ASC",
  "rating": "ASC",
  "stars": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Types

RecruitVarSampFields

aggregate varSamp on columns

Field Name	Description
height - Float	
id - Float	
overallRank - Float	
positionRank - Float	
ranking - Float	
rating - Float	
stars - Float	
weight - Float	
year - Float	

EXAMPLE

```
{  
  "height": 123.45,  
  "id": 123.45,  
  "overallRank": 987.65,  
  "positionRank": 123.45,  
  "ranking": 123.45,  
  "rating": 123.45,  
  "stars": 123.45,  
  "weight": 123.45,  
  "year": 123.45  
}
```

Types

RecruitVarSampOrderBy

order by varSamp() on columns of table
"recruit"

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

EXAMPLE

```
{
  "height": "ASC",
  "id": "ASC",
  "overallRank": "ASC",
  "positionRank": "ASC",
  "ranking": "ASC",
  "rating": "ASC",
  "stars": "ASC",
  "weight": "ASC",
  "year": "ASC"
}
```

Types

RecruitVarianceFields

aggregate variance on columns

Field Name	Description
height - Float	
id - Float	
overallRank - Float	
positionRank - Float	
ranking - Float	
rating - Float	
stars - Float	
weight - Float	
year - Float	

EXAMPLE

```
{  
  "height": 123.45,  
  "id": 123.45,  
  "overallRank": 987.65,  
  "positionRank": 987.65,  
  "ranking": 123.45,  
  "rating": 987.65,  
  "stars": 987.65,  
  "weight": 123.45,  
  "year": 123.45  
}
```

Types

RecruitVarianceOrderBy

order by variance() on columns of table
"recruit"

Input Field	Description
height - OrderBy	
id - OrderBy	
overallRank - OrderBy	
positionRank - OrderBy	
ranking - OrderBy	
rating - OrderBy	
stars - OrderBy	
weight - OrderBy	
year - OrderBy	

EXAMPLE

```
{  
  "height": "ASC",  
  "id": "ASC",  
  "overallRank": "ASC",  
  "positionRank": "ASC",  
  "ranking": "ASC",  
  "rating": "ASC",  
  "stars": "ASC",  
  "weight": "ASC",  
  "year": "ASC"  
}
```

Types

RecruitingTeam

columns and relationships of
"recruiting_team"

Field Name	Description
<code>id</code> - <code>Int!</code>	
<code>points</code> - <code>numeric!</code>	
<code>rank</code> - <code>smallint!</code>	
<code>team</code> - <code>currentTeams</code>	An object relationship
<code>year</code> - <code>smallint!</code>	

EXAMPLE

```
{
  "id": 123,
  "points": numeric,
  "rank": smallint,
  "team": currentTeams,
  "year": smallint
}
```

Types

RecruitingTeamAggregate

aggregated selection of
"recruiting_team"

EXAMPLE

```
{
  "aggregate": RecruitingTeamAggreg
  "nodes": [RecruitingTeam]
}
```

Field Name	Description
aggregate -	
	RecruitingTeamAggregateFields
nodes -	
	[RecruitingTeam!]!

Types

RecruitingTeamAggregateFields

aggregate fields of "recruiting_team"

EXAMPLE

Field Name	Description
avg -	
	RecruitingTeamAvgFields
count - Int!	
	Arguments
columns -	
	[RecruitingTeamSelectColumn!]

```
{
  "avg": RecruitingTeamAvgFields,
  "count": 123,
  "max": RecruitingTeamMaxFields,
  "min": RecruitingTeamMinFields,
  "stddev": RecruitingTeamStddevFie
  "stddevPop": RecruitingTeamStddev
  "stddevSamp": RecruitingTeamStdde
  "sum": RecruitingTeamSumFields,
  "varPop": RecruitingTeamVarPopFie
  "varSamp": RecruitingTeamVarSampF
  "variance": RecruitingTeamVarianc
}
```

Field Name	Description
distinct - Boolean	
max -	RecruitingTeamMaxFields
min -	RecruitingTeamMinFields
stddev -	RecruitingTeamStddevFields
stddevPop -	RecruitingTeamStddevPopFields
stddevSamp -	RecruitingTeamStddevSampFields
sum -	RecruitingTeamSumFields
varPop -	RecruitingTeamVarPopFields
varSamp -	RecruitingTeamVarSampFields
variance -	RecruitingTeamVarianceFields

Types

RecruitingTeamAvgFields

aggregate avg on columns

EXAMPLE

```
{"id": 987.65, "points": 123.45, "r
```

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

Types

RecruitingTeamBoolExp

Boolean expression to filter rows from the table "recruiting_team". All fields are combined with a logical 'AND'.

EXAMPLE

```
{
  "_and": [RecruitingTeamBoolExp],
  "_not": RecruitingTeamBoolExp,
```

Input Field	Description
_and -	
	[RecruitingTeamBoolExp!]
_not -	
	RecruitingTeamBoolExp
_or -	
	[RecruitingTeamBoolExp!]
id -	
	IntComparisonExp
points -	
	NumericComparisonExp
rank -	
	SmallintComparisonExp
team -	
	currentTeamsBoolExp
year -	
	SmallintComparisonExp

```
"_or": [RecruitingTeamBoolExp],  
"id": IntComparisonExp,  
"points": NumericComparisonExp,  
"rank": SmallintComparisonExp,  
"team": currentTeamsBoolExp,  
"year": SmallintComparisonExp  
}
```

Types

RecruitingTeamMaxFields

aggregate max on columns

Field Name	Description
id - Int	
points - numeric	
rank - smallint	
year - smallint	

EXAMPLE

```
{  
  "id": 123,  
  "points": numeric,  
  "rank": smallint,  
  "year": smallint  
}
```

Types

RecruitingTeamMinFields

aggregate min on columns

Field Name	Description
id - Int	
points - numeric	

EXAMPLE

```
{  
  "id": 123,  
  "points": numeric,  
  "rank": smallint,  
  "year": smallint  
}
```

Field Name	Description
rank - smallint	
year - smallint	

Types

RecruitingTeamOrderBy

Ordering options when selecting data from "recruiting_team".

EXAMPLE

```
{  
  "id": "ASC",  
  "points": "ASC",  
  "rank": "ASC",  
  "team": currentTeamsOrderBy,  
  "year": "ASC"  
}
```

Input Field	Description
id - OrderBy	
points - OrderBy	
rank - OrderBy	
team - currentTeamsOrderBy	
year - OrderBy	

Types

RecruitingTeamSelectColumn

select columns of table

"recruiting_team"

EXAMPLE

"id"

Enum Value	Description
id	column name
points	column name
rank	column name
year	column name

Types

RecruitingTeamStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

{"id": 123.45, "points": 123.45, "r

Types

RecruitingTeamStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

{"id": 987.65, "points": 123.45, "r

Types

RecruitingTeamStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

```
{"id": 123.45, "points": 987.65, "r
```

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

Types

RecruitingTeamSumFields

aggregate sum on columns

EXAMPLE

```
{  
  "id": 123,  
  "points": numeric,
```

Field Name	Description	
id - <code>Int</code>		"rank": smallint, "year": smallint }
points - <code>numeric</code>		
rank - <code>smallint</code>		
year - <code>smallint</code>		

Types

RecruitingTeamVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
id - <code>Float</code>	
points - <code>Float</code>	
rank - <code>Float</code>	
year - <code>Float</code>	

{ "id": 987.65, "points": 987.65, "r

Types

RecruitingTeamVarSampFields

aggregate varSamp on columns

EXAMPLE

```
{"id": 987.65, "points": 123.45, "r
```

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

Types

RecruitingTeamVarianceFields

aggregate variance on columns

EXAMPLE

```
{"id": 123.45, "points": 123.45, "r
```

Field Name	Description
id - Float	
points - Float	
rank - Float	
year - Float	

Types

Scoreboard

columns and relationships of "scoreboard"

Field Name	Description
awayClassification - division	
awayConference - String	
awayConferenceAbbreviation - String	

EXAMPLE

```
{
  "awayClassification": division,
  "awayConference": "abc123",
  "awayConferenceAbbreviation": "ab",
  "awayId": 987,
  "awayLineScores": [smallint],
  "awayPoints": smallint,
  "awayTeam": "abc123",
  "city": "xyz789",
  "conferenceGame": true,
  "currentClock": "abc123",
  "currentPeriod": smallint,
  "currentPossession": "xyz789",
  "currentSituation": "abc123",
}
```

Field Name	Description
awayId - <code>Int</code>	
awayLineScores - <code>[smallint!]</code>	
awayPoints - <code>smallint</code>	
awayTeam - <code>String</code>	
city - <code>String</code>	
conferenceGame - <code>Boolean</code>	
currentClock - <code>String</code>	
currentPeriod - <code>smallint</code>	
currentPossession - <code>String</code>	
currentSituation - <code>String</code>	
homeClassification - <code>division</code>	
homeConference - <code>String</code>	
	<pre>"homeClassification": division, "homeConference": "xyz789", "homeConferenceAbbreviation": "xy "homeId": 123, "homeLineScores": [smallint], "homePoints": smallint, "homeTeam": "xyz789", "id": 123, "lastPlay": "abc123", "moneylineAway": 987, "moneylineHome": 987, "neutralSite": true, "overUnder": numeric, "spread": numeric, "startDate": timestamptz, "startTimeTbd": true, "state": "abc123", "status": game_status, "temperature": numeric, "tv": "xyz789", "venue": "xyz789", "weatherDescription": "abc123", "windDirection": numeric, "windSpeed": numeric }</pre>

Field Name	Description
homeConferenceAbbreviation	- <code>String</code>
homeId	- <code>Int</code>
homeLineScores	- <code>[smallint!]</code>
homePoints	- <code>smallint</code>
homeTeam	- <code>String</code>
id	- <code>Int</code>
lastPlay	- <code>String</code>
moneylineAway	- <code>Int</code>
moneylineHome	- <code>Int</code>
neutralSite	- <code>Boolean</code>
overUnder	- <code>numeric</code>
spread	- <code>numeric</code>
startDate	- <code>timestamptz</code>

Field Name	Description
startTimeTbd -	
Boolean	
state -	<code>String</code>
status -	
game_status	
temperature -	
numeric	
tv -	<code>String</code>
venue -	<code>String</code>
weatherDescription	
-	<code>String</code>
windDirection -	
numeric	
windSpeed -	
numeric	

Types

ScoreboardBoolExp

Boolean expression to filter rows from the table "scoreboard". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[ScoreboardBoolExp!]
_not -	ScoreboardBoolExp
_or -	[ScoreboardBoolExp!]
awayClassification	-
awayConference	-
awayConferenceAbbreviation	-
awayId	-
awayLineScores	-

EXAMPLE

```
{
  "_and": [ScoreboardBoolExp],
  "_not": ScoreboardBoolExp,
  "_or": [ScoreboardBoolExp],
  "awayClassification": DivisionComparisonExp,
  "awayConference": StringComparisonExp,
  "awayConferenceAbbreviation": StringComparisonExp,
  "awayId": IntComparisonExp,
  "awayLineScores": SmallintArrayListComparisonExp,
  "awayPoints": SmallintComparisonExp,
  "awayTeam": StringComparisonExp,
  "city": StringComparisonExp,
  "conferenceGame": BooleanComparisonExp,
  "currentClock": StringComparisonExp,
  "currentPeriod": SmallintComparisonExp,
  "currentPossession": StringComparisonExp,
  "currentSituation": StringComparisonExp,
  "homeClassification": DivisionComparisonExp,
  "homeConference": StringComparisonExp,
  "homeConferenceAbbreviation": StringComparisonExp,
  "homeId": IntComparisonExp,
  "homeLineScores": SmallintArrayListComparisonExp,
  "homePoints": SmallintComparisonExp,
  "homeTeam": StringComparisonExp,
  "id": IntComparisonExp,
  "lastPlay": StringComparisonExp,
  "moneylineAway": IntComparisonExp,
  "moneylineHome": IntComparisonExp,
  "neutralSite": BooleanComparisonExp,
  "overUnder": NumericComparisonExp,
  "spread": NumericComparisonExp,
  "startDate": TimestamptzComparisonExp,
  "startTimeTbd": BooleanComparisonExp,
  "state": StringComparisonExp,
  "status": GameStatusComparisonExp,
  "temperature": NumericComparisonExp,
  "tv": StringComparisonExp,
```

Input Field	Description	
awayPoints -		"venue": StringComparisonExp,
SmallintComparisonExp		"weatherDescription": StringCompa
awayTeam -		"windDirection": NumericCompariso
StringComparisonExp		"windSpeed": NumericComparisonExp
city -		}
StringComparisonExp		
conferenceGame -		
BooleanComparisonExp		
currentClock -		
StringComparisonExp		
currentPeriod -		
SmallintComparisonExp		
currentPossession		
-		
StringComparisonExp		
currentSituation		
-		
StringComparisonExp		
homeClassification		
-		
DivisionComparisonExp		
homeConference -		
StringComparisonExp		

Input Field	Description
homeConferenceAbbreviation	-
	StringComparisonExp
homeId	-
	IntComparisonExp
homeLineScores	-
	SmallintArrayComparisonExp
homePoints	-
	SmallintComparisonExp
homeTeam	-
	StringComparisonExp
id	-
	IntComparisonExp
lastPlay	-
	StringComparisonExp
moneylineAway	-
	IntComparisonExp
moneylineHome	-
	IntComparisonExp
neutralSite	-
	BooleanComparisonExp

Input Field	Description
overUnder -	NumericComparisonExp
spread -	NumericComparisonExp
startDate -	TimestamptzComparisonExp
startTimeTbd -	BooleanComparisonExp
state -	StringComparisonExp
status -	GameStatusComparisonExp
temperature -	NumericComparisonExp
tv -	StringComparisonExp
venue -	StringComparisonExp
weatherDescription	-
	StringComparisonExp

Input Field	Description
windDirection -	NumericComparisonExp
windSpeed -	NumericComparisonExp

Types

ScoreboardOrderBy

Ordering options when selecting data from "scoreboard".

Input Field	Description
awayClassification	
- OrderBy	
awayConference	-
	OrderBy
awayConferenceAbbreviation	
- OrderBy	
awayId	- OrderBy

EXAMPLE

```
{
  "awayClassification": "ASC",
  "awayConference": "ASC",
  "awayConferenceAbbreviation": "AS
  "awayId": "ASC",
  "awayLineScores": "ASC",
  "awayPoints": "ASC",
  "awayTeam": "ASC",
  "city": "ASC",
  "conferenceGame": "ASC",
  "currentClock": "ASC",
  "currentPeriod": "ASC",
  "currentPossession": "ASC",
  "currentSituation": "ASC",
  "homeClassification": "ASC",
  "homeConference": "ASC",
```

Input Field	Description
awayLineScores - OrderBy	"homeConferenceAbbreviation": "AS", "homeId": "ASC", "homeLineScores": "ASC", "homePoints": "ASC", "homeTeam": "ASC", "id": "ASC", "lastPlay": "ASC", "moneylineAway": "ASC", "moneylineHome": "ASC", "neutralSite": "ASC", "overUnder": "ASC", "spread": "ASC", "startDate": "ASC", "startTimeTbd": "ASC", "state": "ASC", "status": "ASC", "temperature": "ASC", "tv": "ASC", "venue": "ASC", "weatherDescription": "ASC", "windDirection": "ASC", "windSpeed": "ASC"
awayPoints - OrderBy	}
awayTeam - OrderBy	
city - OrderBy	
conferenceGame - OrderBy	
currentClock - OrderBy	
currentPeriod - OrderBy	
currentPossession - OrderBy	
currentSituation - OrderBy	
homeClassification - OrderBy	
homeConference - OrderBy	
homeConferenceAbbreviation - OrderBy	

Input Field	Description
homeId - OrderBy	
homeLineScores - OrderBy	
homePoints - OrderBy	
homeTeam - OrderBy	
id - OrderBy	
lastPlay - OrderBy	
moneylineAway - OrderBy	
moneylineHome - OrderBy	
neutralSite - OrderBy	
overUnder - OrderBy	
spread - OrderBy	
startDate - OrderBy	
startTimeTbd - OrderBy	

Input Field	Description
state - OrderBy	
status - OrderBy	
temperature - OrderBy	
tv - OrderBy	
venue - OrderBy	
weatherDescription - OrderBy	
windDirection - OrderBy	
windSpeed - OrderBy	

Types

ScoreboardSelectColumn

select columns of table "scoreboard"

[EXAMPLE](#)

Enum Value	Description	
awayClassification	column name	"awayClassification"
awayConference	column name	
awayConferenceAbbr	column name	
awayId	column name	
awayLineScores	column name	
awayPoints	column name	
awayTeam	column name	
city	column name	
conferenceGame	column name	
currentClock	column name	
currentPeriod	column name	
currentPossession	column name	
currentSituation	column name	
homeClassification	column name	
homeConference	column name	
homeConferenceAbbr	column name	
homeId	column name	
homeLineScores	column name	
homePoints	column name	

Enum Value	Description
homeTeam	column name
id	column name
lastPlay	column name
moneylineAway	column name
moneylineHome	column name
neutralSite	column name
overUnder	column name
spread	column name
startDate	column name
startTimeTbd	column name
state	column name
status	column name
temperature	column name
tv	column name
venue	column name
weatherDescription	column name
windDirection	column name
windSpeed	column name

Types

SeasonTypeComparisonExp

Boolean expression to compare columns of type "season_type". All fields are combined with logical 'AND'.

Input Field	Description
_eq - <code>season_type</code>	
_gt - <code>season_type</code>	
_gte - <code>season_type</code>	
_in -	
<code>[season_type!]</code>	
_isNull - <code>Boolean</code>	
_lt - <code>season_type</code>	
_lte - <code>season_type</code>	
_neq - <code>season_type</code>	
_nin -	
<code>[season_type!]</code>	

EXAMPLE

```
{  
  "_eq": season_type,  
  "_gt": season_type,  
  "_gte": season_type,  
  "_in": [season_type],  
  "_isNull": true,  
  "_lt": season_type,  
  "_lte": season_type,  
  "_neq": season_type,  
  "_nin": [season_type]  
}
```

Types

SmallintArrayComparisonExp

Boolean expression to compare columns of type "smallint". All fields are combined with logical 'AND'.

Input Field	Description
_containedIn - [smallint!]	is the array contained in the given array value
_contains - [smallint!]	does the array contain the given value
_eq - [smallint!]	
_gt - [smallint!]	
_gte - [smallint!]	
_in - [smallint!]	
_isNull - Boolean	
_lt - [smallint!]	
_lte - [smallint!]	

EXAMPLE

```
{
  "_containedIn": [smallint],
  "_contains": [smallint],
  "_eq": [smallint],
  "_gt": [smallint],
  "_gte": [smallint],
  "_in": [smallint],
  "_isNull": false,
  "_lt": [smallint],
  "_lte": [smallint],
  "_neq": [smallint],
  "_nin": [smallint]
}
```

Input Field	Description
_neq - [smallint!]	
_nin - [smallint!]	

Types

SmallintComparisonExp

Boolean expression to compare columns
of type "smallint". All fields are
combined with logical 'AND'.

EXAMPLE

```
{
  "_eq": smallint,
  "_gt": smallint,
  "_gte": smallint,
  "_in": [smallint],
  "_isNull": false,
  "_lt": smallint,
  "_lte": smallint,
  "_neq": smallint,
  "_nin": [smallint]
}
```

Input Field	Description
_eq - smallint	
_gt - smallint	
_gte - smallint	
_in - [smallint!]	
_isNull - Boolean	
_lt - smallint	

Input Field	Description
_lte - <small>smallint</small>	
_neq - <small>smallint</small>	
_nin - <small>[smallint!]</small>	

Types

String

EXAMPLE

"xyz789"

Types

StringArrayComparisonExp

Boolean expression to compare columns of type "String". All fields are combined with logical 'AND'.

Input Field	Description
_containedIn - [String!]	is the array contained in the given array value
_contains - [String!]	does the array contain the given value
_eq - [String!]	
_gt - [String!]	
_gte - [String!]	
_in - [String!]	
_isNull - Boolean	
_lt - [String!]	
_lte - [String!]	
_neq - [String!]	
_nin - [String!]	

EXAMPLE

```
{
  "_containedIn": ["xyz789"],
  "_contains": ["xyz789"],
  "_eq": ["abc123"],
  "_gt": ["xyz789"],
  "_gte": ["abc123"],
  "_in": ["abc123"],
  "_isNull": false,
  "_lt": ["xyz789"],
  "_lte": ["xyz789"],
  "_neq": ["xyz789"],
  "_nin": ["abc123"]
}
```

Types

StringComparisonExp

Boolean expression to compare columns of type "String". All fields are combined with logical 'AND'.

Input Field	Description
_eq - String	
_gt - String	
_gte - String	
_ilike - String	does the column match the given case-insensitive pattern
_in - [String!]	does the column match the given
_iregex - String	POSIX regular expression, case insensitive
_isNull - Boolean	

EXAMPLE

```
{
  "_eq": "xyz789",
  "_gt": "xyz789",
  "_gte": "xyz789",
  "_ilike": "abc123",
  "_in": ["xyz789"],
  "_iregex": "xyz789",
  "_isNull": true,
  "_like": "abc123",
  "_lt": "abc123",
  "_lte": "xyz789",
  "_neq": "abc123",
  "_nilike": "abc123",
  "_nin": ["abc123"],
  "_niregex": "xyz789",
  "_nlike": "xyz789",
  "_nregex": "abc123",
  "_nsimilar": "xyz789",
  "_regex": "xyz789",
  "_similar": "abc123"
}
```

Input Field	Description
_like - <code>String</code>	does the column match the given pattern
_lt - <code>String</code>	
_lte - <code>String</code>	
_neq - <code>String</code>	
_nilike - <code>String</code>	does the column NOT match the given case-insensitive pattern
_nin - <code>[String!]</code>	
_niregex - <code>String</code>	does the column NOT match the given POSIX regular expression, case insensitive
_nlike - <code>String</code>	does the column NOT match the given pattern
_nregex - <code>String</code>	does the column NOT match the given POSIX

Input Field	Description
	regular expression, case sensitive
_nsimilar - String	does the column NOT match the given SQL regular expression
_regex - String	does the column match the given POSIX regular expression, case sensitive
_similar - String	does the column match the given SQL regular expression

Types

TeamTalent

columns and relationships of
"team_talent"

Field Name	Description
talent - numeric!	
team - currentTeams	An object relationship
year - smallint!	

EXAMPLE

```
{
  "talent": numeric,
  "team": currentTeams,
  "year": smallint
}
```

Types

TeamTalentAggregate

aggregated selection of "team_talent"

Field Name	Description
aggregate - TeamTalentAggregateFields	
nodes - [TeamTalent!]!	

EXAMPLE

```
{
  "aggregate": TeamTalentAggregateFields,
  "nodes": [TeamTalent]
}
```



Types

TeamTalentAggregateFields

aggregate fields of "team_talent"

Field Name	Description
avg -	TeamTalentAvgFields
count - Int!	
	Arguments
	columns - [TeamTalentSelectColumn!]?
	distinct - Boolean
max -	
	TeamTalentMaxFields
min -	TeamTalentMinFields
stddev -	TeamTalentStddevFields
stddevPop -	TeamTalentStddevPopFields

EXAMPLE

```
{  
  "avg": TeamTalentAvgFields,  
  "count": 123,  
  "max": TeamTalentMaxFields,  
  "min": TeamTalentMinFields,  
  "stddev": TeamTalentStddevFields,  
  "stddevPop": TeamTalentStddevPopF  
  "stddevSamp": TeamTalentStddevSam  
  "sum": TeamTalentSumFields,  
  "varPop": TeamTalentVarPopFields,  
  "varSamp": TeamTalentVarSampField  
  "variance": TeamTalentVarianceFie  
}
```

Field Name	Description
stddevSamp -	TeamTalentStddevSampFields
sum -	TeamTalentSumFields
varPop -	TeamTalentVarPopFields
varSamp -	TeamTalentVarSampFields
variance -	TeamTalentVarianceFields

Types

TeamTalentAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
talent - Float	

```
{"talent": 123.45, "year": 987.65}
```

Field Name	Description
year - Float	

Types

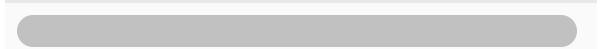
TeamTalentBoolExp

Boolean expression to filter rows from the table "team_talent". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[TeamTalentBoolExp!]
_not -	TeamTalentBoolExp
_or -	[TeamTalentBoolExp!]
talent -	NumericComparisonExp

EXAMPLE

```
{
  "_and": [TeamTalentBoolExp],
  "_not": TeamTalentBoolExp,
  "_or": [TeamTalentBoolExp],
  "talent": NumericComparisonExp,
  "team": currentTeamsBoolExp,
  "year": SmallintComparisonExp
}
```



Input Field	Description
team -	currentTeamsBoolExp
year -	SmallintComparisonExp

Types

TeamTalentMaxFields

aggregate max on columns

EXAMPLE

Field Name	Description
talent - numeric	
year - smallint	

```
{  
  "talent": numeric,  
  "year": smallint  
}
```

Types

TeamTalentMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
talent - numeric	
year - smallint	

```
{  
  "talent": numeric,  
  "year": smallint  
}
```

Types

TeamTalentOrderBy

Ordering options when selecting data from "team_talent".

EXAMPLE

Input Field	Description
talent - OrderBy	
team - currentTeamsOrderBy	

```
{  
  "talent": "ASC",  
  "team": currentTeamsOrderBy,  
  "year": "ASC"  
}
```

Input Field	Description
year - OrderBy	

Types

TeamTalentSelectColumn

select columns of table "team_talent"

[EXAMPLE](#)

Enum Value	Description	"talent"
talent	column name	
year	column name	

Types

TeamTalentStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

{ "talent": 123.45, "year": 123.45 }

Types

TeamTalentStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

{ "talent": 987.65, "year": 987.65 }

Types

TeamTalentStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

```
{"talent": 123.45, "year": 987.65}
```

Types

TeamTalentSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
talent - numeric	
year - smallint	

```
{
  "talent": numeric,
  "year": smallint
}
```

Types

TeamTalentVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

```
{"talent": 123.45, "year": 123.45}
```

Types

TeamTalentVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

```
{"talent": 987.65, "year": 123.45}
```

Types

TeamTalentVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
talent - Float	
year - Float	

```
{"talent": 123.45, "year": 987.65}
```

Types

TimestampComparisonExp

Boolean expression to compare columns
of type "timestamp". All fields are
combined with logical 'AND'.

EXAMPLE

```
{  
  "_eq": timestamp,  
  "_gt": timestamp,  
  "_gte": timestamp,
```

Input Field	Description
_eq - timestamp	
_gt - timestamp	
_gte - timestamp	
_in - [timestamp!]	
_isNull - Boolean	
_lt - timestamp	
_lte - timestamp	
_neq - timestamp	
_nin - [timestamp!]	

```

    "_in": [timestamp],
    "_isNull": true,
    "_lt": timestamp,
    "_lte": timestamp,
    "_neq": timestamp,
    "_nin": [timestamp]
}
```

Types

TimestamptzComparisonExp

Boolean expression to compare columns of type "timestamptz". All fields are combined with logical 'AND'.

EXAMPLE

```
{
  "_eq": timestamptz,
  "_gt": timestamptz,
```

Input Field	Description
_eq - timestamptz	
_gt - timestamptz	
_gte - timestamptz	
_in - [timestamptz!]	
_isNull - Boolean	
_lt - timestamptz	
_lte - timestamptz	
_neq - timestamptz	
_nin - [timestamptz!]	

```

    "_gte": timestamptz,
    "_in": [timestamptz],
    "_isNull": true,
    "_lt": timestamptz,
    "_lte": timestamptz,
    "_neq": timestamptz,
    "_nin": [timestamptz]
}
```

Types

Transfer

columns and relationships of "transfer"

EXAMPLE

```
{
  "eligibility": "xyz789",
```

Field Name	Description	
eligibility - String		"firstName": "xyz789", "fromTeam": currentTeams, "lastName": "xyz789", "position": RecruitPosition, "rating": numeric, "season": smallint, "stars": smallint, "toTeam": currentTeams, "transferDate": timestamp
firstName - String!		}
fromTeam - currentTeams	An object relationship	
lastName - String!		
position - RecruitPosition	An object relationship	
rating - numeric		
season - smallint!		
stars - smallint		
toTeam - currentTeams	An object relationship	
transferDate - timestamp		

Types

TransferBoolExp

Boolean expression to filter rows from the table "transfer". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[TransferBoolExp!]
_not -	TransferBoolExp
_or -	[TransferBoolExp!]
eligibility -	StringComparisonExp
firstName -	StringComparisonExp
fromTeam -	currentTeamsBoolExp
lastName -	StringComparisonExp
position -	RecruitPositionBoolExp

EXAMPLE

```
{  
  "_and": [TransferBoolExp],  
  "_not": TransferBoolExp,  
  "_or": [TransferBoolExp],  
  "eligibility": StringComparisonExp,  
  "firstName": StringComparisonExp,  
  "fromTeam": currentTeamsBoolExp,  
  "lastName": StringComparisonExp,  
  "position": RecruitPositionBoolExp,  
  "rating": NumericComparisonExp,  
  "season": SmallintComparisonExp,  
  "stars": SmallintComparisonExp,  
  "toTeam": currentTeamsBoolExp,  
  "transferDate": TimestampComparis  
}
```

Input Field	Description
rating -	NumericComparisonExp
season -	SmallintComparisonExp
stars -	SmallintComparisonExp
toTeam -	currentTeamsBoolExp
transferDate -	TimestampComparisonExp

Types

TransferOrderBy

Ordering options when selecting data from "transfer".

EXAMPLE

```
{  
  "eligibility": "ASC",  
  "firstName": "ASC",  
  "fromTeam": currentTeamsOrderBy,  
  "lastName": "ASC",  
}
```

Input Field	Description	
eligibility - OrderBy		"position": RecruitPositionOrderBy
firstName - OrderBy		"rating": "ASC",
fromTeam - currentTeamsOrderBy		"season": "ASC",
lastName - OrderBy		"stars": "ASC",
position - RecruitPositionOrderBy		"toTeam": currentTeamsOrderBy,
rating - OrderBy		"transferDate": "ASC"
season - OrderBy		}
stars - OrderBy		
toTeam - currentTeamsOrderBy		
transferDate - OrderBy		

Types

TransferSelectColumn

select columns of table "transfer"

EXAMPLE

Enum Value	Description	
eligibility	column name	"eligibility"
firstName	column name	
lastName	column name	
rating	column name	
season	column name	
stars	column name	
transferDate	column name	

Types

WeatherCondition

columns and relationships of
"weather_condition"

EXAMPLE

Field Name	Description
description - String!	
id - smallint!	

```
{
  "description": "xyz789",
  "id": smallint
}
```

Types

WeatherConditionBoolExp

Boolean expression to filter rows from the table "weather_condition". All fields are combined with a logical 'AND'.

Input Field	Description
_and - [WeatherConditionBoolExp!]	
_not - WeatherConditionBoolExp	
_or - [WeatherConditionBoolExp!]	

EXAMPLE

```
{
  "_and": [WeatherConditionBoolExp]
  "_not": WeatherConditionBoolExp,
  "_or": [WeatherConditionBoolExp],
  "description": StringComparisonExp
  "id": SmallintComparisonExp
}
```

Input Field	Description
description -	
	StringComparisonExp
id -	
	SmallintComparisonExp

Types

WeatherConditionOrderBy

Ordering options when selecting data from "weather_condition".

EXAMPLE

```
{"description": "ASC", "id": "ASC"}
```

Input Field	Description
description -	
	OrderBy
id -	OrderBy

Types

WeatherConditionSelectColumn

select columns of table
"weather_condition"

EXAMPLE

"description"

Enum Value	Description
description	column name
id	column name

Types

adjustedPlayerMetricsAggregateBo...

Input Field	Description
arguments -	AdjustedPlayerMetricsSelectColumn!
distinct - Boolean	
filter -	AdjustedPlayerMetricsBoolExp

EXAMPLE

```
{  
  "arguments": ["athleteId"],  
  "distinct": false,  
  "filter": AdjustedPlayerMetricsBo  
  "predicate": IntComparisonExp  
}
```

Input Field	Description
predicate - IntComparisonExp!	

Types

athleteAggregateBoolExpCount

Input Field	Description
arguments - [AthleteSelectColumn!]	
distinct - Boolean	
filter - AthleteBoolExp	
predicate - IntComparisonExp!	

EXAMPLE

```
{  
  "arguments": ["firstName"],  
  "distinct": false,  
  "filter": AthleteBoolExp,  
  "predicate": IntComparisonExp  
}
```

Types

athleteTeamAggregateBoolExpCount

Input Field	Description	EXAMPLE
arguments -		
	[AthleteTeamSelectColumn!]	
distinct -	Boolean	
filter -		
	AthleteTeamBoolExp	
predicate -		
	IntComparisonExp!	

Types

bigint

EXAMPLE

bigint

Types

coachSeasonAggregateBoolExpCo...

Input Field	Description	EXAMPLE
arguments -		{
CoachSeasonSelectColumn!		"arguments": ["games"],
distinct - Boolean		"distinct": true,
filter -		"filter": CoachSeasonBoolExp,
CoachSeasonBoolExp		"predicate": IntComparisonExp
predicate -		}
IntComparisonExp!		

Types

currentTeams

columns and relationships of
"current_conferences"

EXAMPLE

```
{  
    "abbreviation": "xyz789",
```

Field Name	Description	
abbreviation - String		"classification": division, "conference": "abc123", "conferenceId": smallint, "division": "xyz789", "school": "abc123", "teamId": 987
classification - division		}
conference - String		
conferenceId - smallint		
division - String		
school - String		
teamId - Int		

Types

currentTeamsAggregate

aggregated selection of
"current_conferences"

EXAMPLE

```
{
  "aggregate": currentTeamsAggregat
```

Field Name	Description
aggregate -	
currentTeamsAggregateFields	
nodes -	
[currentTeams!]!	

```
"nodes": [currentTeams]
```

```
}
```

Types

currentTeamsAggregateFields

aggregate fields of
"current_conferences"

Field Name	Description
avg -	
currentTeamsAvgFields	
count - Int!	

Arguments

columns - [\[currentTeamsSelectColumn!\]](#)

EXAMPLE

```
{
  "avg": currentTeamsAvgFields,
  "count": 123,
  "max": currentTeamsMaxFields,
  "min": currentTeamsMinFields,
  "stddev": currentTeamsStddevField
  "stddevPop": currentTeamsStddevPo
  "stddevSamp": currentTeamsStddevS
  "sum": currentTeamsSumFields,
  "varPop": currentTeamsVarPopField
  "varSamp": currentTeamsVarSampFie
  "variance": currentTeamsVarianceF}
```

```
}
```

Field Name	Description
------------	-------------

distinct - Boolean	
--------------------	--

max -	
-------	--

[currentTeamsMaxFields](#)

min -	
-------	--

[currentTeamsMinFields](#)

stddev -	
----------	--

[currentTeamsStddevFields](#)

stddevPop -	
-------------	--

[currentTeamsStddevPopFields](#)

stddevSamp -	
--------------	--

[currentTeamsStddevSampFields](#)

sum -	
-------	--

[currentTeamsSumFields](#)

varPop -	
----------	--

[currentTeamsVarPopFields](#)

varSamp -	
-----------	--

[currentTeamsVarSampFields](#)

variance -	
------------	--

[currentTeamsVarianceFields](#)

Types

currentTeamsAvgFields

aggregate avg on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId -	Float

```
{"conferenceId": 123.45, "teamId":
```



Types

currentTeamsBoolExp

Boolean expression to filter rows from the table "current_conferences". All fields are combined with a logical 'AND'.

EXAMPLE

```
{
  "_and": [currentTeamsBoolExp],
  "_not": currentTeamsBoolExp,
  "_or": [currentTeamsBoolExp],
  "abbreviation": StringComparisonE
  "classification": DivisionCompa
```

Input Field	Description	
_and -		"conference": StringComparisonExp
[currentTeamsBoolExp!]		"conferenceId": SmallintCompariso
_not -		"division": StringComparisonExp,
currentTeamsBoolExp		"school": StringComparisonExp,
_or -		"teamId": IntComparisonExp
[currentTeamsBoolExp!]		}
abbreviation -		
StringComparisonExp		
classification -		
DivisionComparisonExp		
conference -		
StringComparisonExp		
conferenceId -		
SmallintComparisonExp		
division -		
StringComparisonExp		
school -		
StringComparisonExp		
teamId -		
IntComparisonExp		

Types

currentTeamsMaxFields

aggregate max on columns

Field Name	Description
abbreviation - String	
classification - division	
conference - String	
conferenceId - smallint	
division - String	
school - String	
teamId - Int	

EXAMPLE

```
{  
  "abbreviation": "abc123",  
  "classification": division,  
  "conference": "abc123",  
  "conferenceId": smallint,  
  "division": "xyz789",  
  "school": "xyz789",  
  "teamId": 987  
}
```

Types

currentTeamsMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
abbreviation - String	
classification - division	
conference - String	
conferenceId - smallint	
division - String	
school - String	
teamId - Int	

```
{  
  "abbreviation": "abc123",  
  "classification": division,  
  "conference": "xyz789",  
  "conferenceId": smallint,  
  "division": "xyz789",  
  "school": "xyz789",  
  "teamId": 987  
}
```

Types

currentTeamsOrderBy

Ordering options when selecting data from "current_conferences".

Input Field	Description
abbreviation - OrderBy	
classification - OrderBy	
conference - OrderBy	
conferenceId - OrderBy	
division - OrderBy	
school - OrderBy	
teamId - OrderBy	

EXAMPLE

```
{  
  "abbreviation": "ASC",  
  "classification": "ASC",  
  "conference": "ASC",  
  "conferenceId": "ASC",  
  "division": "ASC",  
  "school": "ASC",  
  "teamId": "ASC"  
}
```

Types

currentTeamsSelectColumn

select columns of table
"current_conferences"

EXAMPLE

"abbreviation"

Enum Value	Description
abbreviation	column name
classification	column name
conference	column name
conferenceId	column name
division	column name
school	column name
teamId	column name

Types

currentTeamsStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId - Float	

```
{"conferenceId": 123.45, "teamId":
```

Types

currentTeamsStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId - Float	

```
{"conferenceId": 123.45, "teamId":
```

Types

currentTeamsStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId -	Float

```
{"conferenceId": 987.65, "teamId":
```

Types

currentTeamsSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId -	Int

```
{"conferenceId": smallint, "teamId":
```

Types

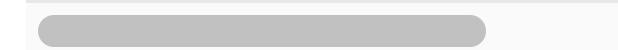
currentTeamsVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId -	Float

```
{"conferenceId": 987.65, "teamId":
```



Types

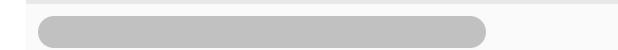
currentTeamsVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
conferenceId -	
teamId -	Float

```
{"conferenceId": 987.65, "teamId":
```



Field Name	Description
teamId - Float	

Types

currentTeamsVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
conferenceId - Float	
teamId - Float	

{ "conferenceId": 987.65, "teamId":

Types

division

EXAMPLE`division`

Types

draftPicksAggregateBoolExpCount

Input Field	Description
arguments -	
[DraftPicksSelectColumn!]	
distinct - Boolean	
filter -	
DraftPicksBoolExp	
predicate -	
IntComparisonExp!	

EXAMPLE

```
{  
  "arguments": ["collegeId"],  
  "distinct": false,  
  "filter": DraftPicksBoolExp,  
  "predicate": IntComparisonExp  
}
```

Types

game

columns and relationships of
"game_info"

Field Name	Description
attendance - Int	
awayClassification - division	
awayConference - String	
awayConferenceId - smallint	
awayConferenceInfo - Conference	An object relationship
awayEndElo - Int	
awayLineScores - [smallint!]	
awayPoints - smallint	
awayPostgameWinProb - numeric	

EXAMPLE

```
{
  "attendance": 987,
  "awayClassification": division,
  "awayConference": "abc123",
  "awayConferenceId": smallint,
  "awayConferenceInfo": Conference,
  "awayEndElo": 987,
  "awayLineScores": [smallint],
  "awayPoints": smallint,
  "awayPostgameWinProb": numeric,
  "awayStartElo": 987,
  "awayTeam": "abc123",
  "awayTeamId": 987,
  "awayTeamInfo": currentTeams,
  "conferenceGame": false,
  "excitement": numeric,
  "homeClassification": division,
  "homeConference": "abc123",
  "homeConferenceId": smallint,
  "homeConferenceInfo": Conference,
  "homeEndElo": 123,
  "homeLineScores": [smallint],
  "homePoints": smallint,
  "homePostgameWinProb": numeric,
  "homeStartElo": 123,
  "homeTeam": "xyz789",
  "homeTeamId": 987,
  "homeTeamInfo": currentTeams,
  "id": 123,
  "lines": [GameLines],
  "linesAggregate": GameLinesAggreg,
  "mediaInfo": [GameMedia],
```

Field Name	Description
awayStartElo - <code>Int</code>	
awayTeam - <code>String</code>	
awayTeamId - <code>Int</code>	
awayTeamInfo - <code>currentTeams</code>	An object relationship
conferenceGame - <code>Boolean</code>	
excitement - <code>numeric</code>	
homeClassification - <code>division</code>	
homeConference - <code>String</code>	
homeConferenceId - <code>smallint</code>	
homeConferenceInfo - <code>Conference</code>	An object relationship
homeEndElo - <code>Int</code>	
homeLineScores - <code>[smallint!]</code>	

```

"neutralSite": true,
"notes": "abc123",
"season": smallint,
"seasonType": season_type,
"startDate": timestamp,
"startTimeTbd": false,
"status": game_status,
"venueId": 987,
"weather": GameWeather,
"week": smallint
}
}
```

Field Name	Description
homePoints - smallint	
homePostgameWinProb - numeric	
homeStartElo - Int	
homeTeam - String	
homeTeamId - Int	
homeTeamInfo - currentTeams	An object relationship
id - Int	
lines - [GameLines!]!	An array relationship
Arguments	
distinctOn - [GameLinesSelectColumn!]	distinct select on columns
limit - Int	limit the number of rows returned
offset - Int	skip the first n rows. Use only with order_by

Field Name	Description
------------	-------------

orderBy - [GameLinesOrderBy!]	
---	--

sort the rows by one or more columns

where - GameLinesBoolExp	
--	--

filter the rows returned

linesAggregate - An aggregate	
-------------------------------	--

[GameLinesAggregate](#) relationship

Arguments

distinctOn - [GameLinesSelectColumn!]	
---	--

distinct select on columns

limit - Int	
-----------------------------	--

limit the number of rows returned

offset - Int	
------------------------------	--

skip the first n rows. Use only with
order_by

orderBy - [GameLinesOrderBy!]	
---	--

sort the rows by one or more columns

where - GameLinesBoolExp	
--	--

filter the rows returned

Field Name	Description
------------	-------------

mediaInfo - [GameMedia!]!	An array relationship
---------------------------	-----------------------

Arguments

distinctOn - [GameMediaSelectColumn!]

distinct select on columns

limit - Int

limit the number of rows returned

offset - Int

skip the first n rows. Use only with order_by

orderBy - [GameMediaOrderBy!]

sort the rows by one or more columns

where - GameMediaBoolExp

filter the rows returned

neutralSite - Boolean

notes - String

season - smallint

Field Name	Description
seasonType -	
season_type	
startDate -	
timestamp	
startTimeTbd -	
Boolean	
status -	
game_status	
venueId - Int	
weather -	An object
GameWeather	relationship
week - smallint	

Types

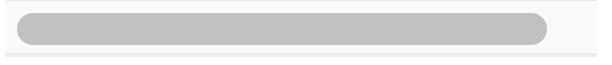
gameAggregate

aggregated selection of "game_info"

EXAMPLE

Field Name	Description
aggregate -	
gameAggregateFields	
nodes - [game!]!	

```
{
  "aggregate": gameAggregateFields,
  "nodes": [game]
}
```



Types

gameAggregateFields

aggregate fields of "game_info"

EXAMPLE

Field Name	Description
avg -	
gameAvgFields	
count - Int!	

Arguments

columns - [gameSelectColumn!]
distinct - Boolean

```
{
  "avg": gameAvgFields,
  "count": 123,
  "max": gameMaxFields,
  "min": gameMinFields,
  "stddev": gameStddevFields,
  "stddevPop": gameStddevPopFields,
  "stddevSamp": gameStddevSampField
  "sum": gameSumFields,
  "varPop": gameVarPopFields,
  "varSamp": gameVarSampFields,
  "variance": gameVarianceFields
}
```



Field Name	Description
max -	
	gameMaxFields
min -	
	gameMinFields
stddev -	
	gameStddevFields
stddevPop -	
	gameStddevPopFields
stddevSamp -	
	gameStddevSampFields
sum -	
	gameSumFields
varPop -	
	gameVarPopFields
varSamp -	
	gameVarSampFields
variance -	
	gameVarianceFields

Types

gameAvgFields

aggregate avg on columns

Field Name	Description
attendance - <code>Float</code>	
awayConferenceId - <code>Float</code>	
awayEndElo - <code>Float</code>	
awayPoints - <code>Float</code>	
awayPostgameWinProb - <code>Float</code>	
awayStartElo - <code>Float</code>	
awayTeamId - <code>Float</code>	
excitement - <code>Float</code>	
homeConferenceId - <code>Float</code>	
homeEndElo - <code>Float</code>	
homePoints - <code>Float</code>	

EXAMPLE

```
{  
  "attendance": 987.65,  
  "awayConferenceId": 123.45,  
  "awayEndElo": 987.65,  
  "awayPoints": 123.45,  
  "awayPostgameWinProb": 123.45,  
  "awayStartElo": 123.45,  
  "awayTeamId": 123.45,  
  "excitement": 987.65,  
  "homeConferenceId": 123.45,  
  "homeEndElo": 123.45,  
  "homePoints": 987.65,  
  "homePostgameWinProb": 123.45,  
  "homeStartElo": 123.45,  
  "homeTeamId": 123.45,  
  "id": 987.65,  
  "season": 987.65,  
  "venueId": 987.65,  
  "week": 987.65  
}
```

Field Name	Description
homePostgameWinProb	- Float
homeStartElo	- Float
homeTeamId	- Float
id	- Float
season	- Float
venueId	- Float
week	- Float

Types

gameBoolExp

Boolean expression to filter rows from the table "game_info". All fields are combined with a logical 'AND'.

EXAMPLE

```
{  
  "_and": [gameBoolExp],  
  "_not": gameBoolExp,  
  "_or": [gameBoolExp],  
  "attendance": IntComparisonExp,  
  "awayClassification": DivisionCom
```

Input Field	Description
_and -	
[gameBoolExp!]	
_not - gameBoolExp	
_or -	
[gameBoolExp!]	
attendance -	
IntComparisonExp	
awayClassification	
-	
DivisionComparisonExp	
awayConference -	
StringComparisonExp	
awayConferenceId	
-	
SmallintComparisonExp	
awayConferenceInfo	
-	
ConferenceBoolExp	
awayEndElo -	
IntComparisonExp	
awayLineScores -	
SmallintArrayComparisonExp	

"awayConference": StringComparisonExp,
 "awayConferenceId": SmallintComparisonExp,
 "awayConferenceInfo": ConferenceBooleanComparisonExp,
 "awayEndElo": IntComparisonExp,
 "awayLineScores": SmallintArrayComparisonExp,
 "awayPoints": SmallintComparisonExp,
 "awayPostgameWinProb": NumericComparisonExp,
 "awayStartElo": IntComparisonExp,
 "awayTeam": StringComparisonExp,
 "awayTeamId": IntComparisonExp,
 "awayTeamInfo": currentTeamsBooleanComparisonExp,
 "conferenceGame": BooleanComparisonExp,
 "excitement": NumericComparisonExp,
 "homeClassification": DivisionComparisonExp,
 "homeConference": StringComparisonExp,
 "homeConferenceId": SmallintComparisonExp,
 "homeConferenceInfo": ConferenceBooleanComparisonExp,
 "homeEndElo": IntComparisonExp,
 "homeLineScores": SmallintArrayComparisonExp,
 "homePoints": SmallintComparisonExp,
 "homePostgameWinProb": NumericComparisonExp,
 "homeStartElo": IntComparisonExp,
 "homeTeam": StringComparisonExp,
 "homeTeamId": IntComparisonExp,
 "homeTeamInfo": currentTeamsBooleanComparisonExp,
 "id": IntComparisonExp,
 "lines": GameLinesBoolExp,
 "linesAggregate": GameLinesAggregateComparisonExp,
 "mediaInfo": GameMediaBoolExp,
 "neutralSite": BooleanComparisonExp,
 "notes": StringComparisonExp,
 "season": SmallintComparisonExp,
 "seasonType": SeasonTypeComparisonExp,
 "startDate": TimestampComparisonExp,
 "startTimeTbd": BooleanComparisonExp,
 "status": GameStatusComparisonExp,
 "venueId": IntComparisonExp,
 "weather": GameWeatherBoolExp,

Input Field	Description	
awayPoints -		"week": SmallintComparisonExp
	SmallintComparisonExp	}
awayPostgameWinProb	-	
	NumericComparisonExp	
awayStartElo -		
	IntComparisonExp	
awayTeam -		
	StringComparisonExp	
awayTeamId -		
	IntComparisonExp	
awayTeamInfo -		
	currentTeamsBoolExp	
conferenceGame -		
	BooleanComparisonExp	
excitement -		
	NumericComparisonExp	
homeClassification	-	
	DivisionComparisonExp	
homeConference -		
	StringComparisonExp	

Input Field	Description
-------------	-------------

homeConferenceId	
------------------	--

-

	SmallintComparisonExp
--	---------------------------------------

homeConferenceInfo	
--------------------	--

-

	ConferenceBoolExp
--	-----------------------------------

homeEndElo	-
------------	---

	IntComparisonExp
--	----------------------------------

homeLineScores	-
----------------	---

	SmallIntArrayComparisonExp
--	--

homePoints	-
------------	---

	SmallintComparisonExp
--	---------------------------------------

homePostgameWinProb	
---------------------	--

-

	NumericComparisonExp
--	--------------------------------------

homeStartElo	-
--------------	---

	IntComparisonExp
--	----------------------------------

homeTeam	-
----------	---

	StringComparisonExp
--	-------------------------------------

homeTeamId	-
------------	---

	IntComparisonExp
--	----------------------------------

homeTeamInfo	-
--------------	---

	currentTeamsBoolExp
--	-------------------------------------

Input Field	Description
id -	IntComparisonExp
lines -	GameLinesBoolExp
linesAggregate -	GameLinesAggregateBoolExp
mediaInfo -	GameMediaBoolExp
neutralSite -	BooleanComparisonExp
notes -	StringComparisonExp
season -	SmallintComparisonExp
seasonType -	SeasonTypeComparisonExp
startDate -	TimestampComparisonExp
startTimeTbd -	BooleanComparisonExp
status -	GameStatusComparisonExp

Input Field	Description
venueId -	
	IntComparisonExp
weather -	
	GameWeatherBoolExp
week -	
	SmallintComparisonExp

Types

gameLinesAggregateBoolExpCount

Input Field	Description	EXAMPLE
arguments -		
	[GameLinesSelectColumn!]	{ "arguments": ["gameId"], "distinct": false, "filter": GameLinesBoolExp, "predicate": IntComparisonExp }
distinct - Boolean		
filter -		
	GameLinesBoolExp	
predicate -		
	IntComparisonExp!	

Types

gameMaxFields

aggregate max on columns

Field Name	Description
attendance - Int	
awayClassification - division	
awayConference - String	
awayConferenceId - smallint	
awayEndElo - Int	
awayLineScores - [smallint!]	
awayPoints - smallint	

EXAMPLE

```
{  
  "attendance": 123,  
  "awayClassification": division,  
  "awayConference": "xyz789",  
  "awayConferenceId": smallint,  
  "awayEndElo": 987,  
  "awayLineScores": [smallint],  
  "awayPoints": smallint,  
  "awayPostgameWinProb": numeric,  
  "awayStartElo": 987,  
  "awayTeam": "xyz789",  
  "awayTeamId": 987,  
  "excitement": numeric,  
  "homeClassification": division,  
  "homeConference": "xyz789",  
  "homeConferenceId": smallint,  
  "homeEndElo": 987,  
  "homeLineScores": [smallint],  
  "homePoints": smallint,  
  "homePostgameWinProb": numeric,  
  "homeStartElo": 987,  
  "homeTeam": "abc123",  
  "homeTeamId": 987,  
  "id": 987,  
  "notes": "abc123",  
}
```

Field Name	Description
awayPostgameWinProb	- numeric
awayStartElo	- Int
awayTeam	- String
awayTeamId	- Int
excitement	- numeric
homeClassification	- division
homeConference	- String
homeConferenceId	- smallint
homeEndElo	- Int
homeLineScores	- [smallint!]
homePoints	- smallint
homePostgameWinProb	- numeric
homeStartElo	- Int

```

    "season": smallint,
    "seasonType": season_type,
    "startDate": timestamp,
    "status": game_status,
    "venueId": 987,
    "week": smallint
}

```

Field Name	Description
homeTeam - <code>String</code>	
homeTeamId - <code>Int</code>	
id - <code>Int</code>	
notes - <code>String</code>	
season - <code>smallint</code>	
seasonType - <code>season_type</code>	
startDate - <code>timestamp</code>	
status - <code>game_status</code>	
venueId - <code>Int</code>	
week - <code>smallint</code>	

Types

gameMinFields

aggregate min on columns

Field Name	Description
attendance - <code>Int</code>	
awayClassification - <code>division</code>	
awayConference - <code>String</code>	
awayConferenceId - <code>smallint</code>	
awayEndElo - <code>Int</code>	
awayLineScores - <code>[smallint!]</code>	
awayPoints - <code>smallint</code>	
awayPostgameWinProb - <code>numeric</code>	
awayStartElo - <code>Int</code>	
awayTeam - <code>String</code>	
awayTeamId - <code>Int</code>	
excitement - <code>numeric</code>	

EXAMPLE

```
{
  "attendance": 987,
  "awayClassification": division,
  "awayConference": "abc123",
  "awayConferenceId": smallint,
  "awayEndElo": 987,
  "awayLineScores": [smallint],
  "awayPoints": smallint,
  "awayPostgameWinProb": numeric,
  "awayStartElo": 987,
  "awayTeam": "xyz789",
  "awayTeamId": 123,
  "excitement": numeric,
  "homeClassification": division,
  "homeConference": "xyz789",
  "homeConferenceId": smallint,
  "homeEndElo": 987,
  "homeLineScores": [smallint],
  "homePoints": smallint,
  "homePostgameWinProb": numeric,
  "homeStartElo": 987,
  "homeTeam": "abc123",
  "homeTeamId": 123,
  "id": 987,
  "notes": "xyz789",
  "season": smallint,
  "seasonType": season_type,
  "startDate": timestamp,
  "status": game_status,
  "venueId": 987,
  "week": smallint
}
```

Field Name	Description
homeClassification	
- division	
homeConference	-
	String
homeConferenceId	
- smallint	
homeEndElo	- Int
homeLineScores	-
	[smallint!]
homePoints	-
	smallint
homePostgameWinProb	
- numeric	
homeStartElo	- Int
homeTeam	- String
homeTeamId	- Int
id	- Int
notes	- String
season	- smallint
seasonType	-
	season_type

Field Name	Description
startDate -	
timestamp	
status -	
game_status	
venueId -	Int
week -	smallint

Types

gameOrderBy

Ordering options when selecting data from "game_info".

Input Field	Description
attendance -	
OrderBy	
awayClassification	
- OrderBy	

EXAMPLE

```
{
  "attendance": "ASC",
  "awayClassification": "ASC",
  "awayConference": "ASC",
  "awayConferenceId": "ASC",
  "awayConferenceInfo": Conference0
  "awayEndElo": "ASC",
  "awayLineScores": "ASC",
  "awayPoints": "ASC",
  "awayPostgameWinProb": "ASC",
  "awayStartElo": "ASC",
  "awayTeam": "ASC",
```

Input Field	Description
awayConference - OrderBy	"awayTeamId": "ASC", "awayTeamInfo": currentTeamsOrderBy "conferenceGame": "ASC", "excitement": "ASC", "homeClassification": "ASC", "homeConference": "ASC", "homeConferenceId": "ASC", "homeConferenceInfo": ConferenceOrderBy "homeEndElo": "ASC", "homeLineScores": "ASC", "homePoints": "ASC", "homePostgameWinProb": "ASC", "homeStartElo": "ASC", "homeTeam": "ASC", "homeTeamId": "ASC", "homeTeamInfo": currentTeamsOrderBy "id": "ASC", "linesAggregate": GameLinesAggregate, "mediaInfoAggregate": GameMediaAggregate, "neutralSite": "ASC", "notes": "ASC", "season": "ASC", "seasonType": "ASC", "startDate": "ASC", "startTimeTbd": "ASC", "status": "ASC", "venueId": "ASC", "weather": GameWeatherOrderBy, "week": "ASC"
awayConferenceId - OrderBy	
awayConferenceInfo - ConferenceOrderBy	
awayEndElo - OrderBy	
awayLineScores - OrderBy	
awayPoints - OrderBy	
awayPostgameWinProb - OrderBy	
awayStartElo - OrderBy	
awayTeam - OrderBy	
awayTeamId - OrderBy	
awayTeamInfo - currentTeamsOrderBy	

Input Field	Description
conferenceGame -	
OrderBy	
excitement -	
OrderBy	
homeClassification	
- OrderBy	
homeConference -	
OrderBy	
homeConferenceId	
- OrderBy	
homeConferenceInfo	
-	
ConferenceOrderBy	
homeEndElo -	
OrderBy	
homeLineScores -	
OrderBy	
homePoints -	
OrderBy	
homePostgameWinProb	
- OrderBy	

Input Field	Description
homeStartElo -	
OrderBy	
homeTeam -	OrderBy
homeTeamId -	
OrderBy	
homeTeamInfo -	
currentTeamsOrderBy	
id -	OrderBy
linesAggregate -	
GameLinesAggregateOrderBy	
mediaInfoAggregate	
-	
GameMediaAggregateOrderBy	
neutralSite -	
OrderBy	
notes -	OrderBy
season -	OrderBy
seasonType -	
OrderBy	
startDate -	
OrderBy	

Input Field	Description
startTimeTbd -	
status - OrderBy	
venueId - OrderBy	
weather -	
week - OrderBy	

Types

gamePlayerStatAggregateBoolExpC...

Input Field	Description
arguments -	
[GamePlayerStatSelectColumn!]	
distinct - Boolean	
filter -	
GamePlayerStatBoolExp	

EXAMPLE

```
{
  "arguments": ["athleteId"],
  "distinct": false,
  "filter": GamePlayerStatBoolExp,
  "predicate": IntComparisonExp
}
```

Input Field	Description
<code>predicate - IntComparisonExp!</code>	
<hr/>	
Types	

gameSelectColumn

select columns of table "game_info"

EXAMPLE

`"attendance"`

Enum Value	Description
<code>attendance</code>	column name
<code>awayClassification</code>	column name
<code>awayConference</code>	column name
<code>awayConferenceId</code>	column name
<code>awayEndElo</code>	column name
<code>awayLineScores</code>	column name
<code>awayPoints</code>	column name
<code>awayPostgameWinProb</code>	column name

Enum Value	Description
awayStartElo	column name
awayTeam	column name
awayTeamId	column name
conferenceGame	column name
excitement	column name
homeClassification	column name
homeConference	column name
homeConferenceId	column name
homeEndElo	column name
homeLineScores	column name
homePoints	column name
homePostgameWinProb	column name
homeStartElo	column name
homeTeam	column name
homeTeamId	column name
id	column name
neutralSite	column name
notes	column name
season	column name

Enum Value	Description
seasonType	column name
startDate	column name
startTimeTbd	column name
status	column name
venueId	column name
week	column name

Types

gameStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
attendance - Float	
awayConferenceId - Float	
awayEndElo - Float	

```
{
  "attendance": 987.65,
  "awayConferenceId": 987.65,
  "awayEndElo": 987.65,
  "awayPoints": 987.65,
  "awayPostgameWinProb": 987.65,
  "awayStartElo": 123.45,
  "awayTeamId": 123.45,
  "excitement": 123.45,
  "homeConferenceId": 123.45,
```

Field Name	Description
awayPoints - <code>Float</code>	
awayPostgameWinProb - <code>Float</code>	
awayStartElo - <code>Float</code>	
awayTeamId - <code>Float</code>	
excitement - <code>Float</code>	
homeConferenceId - <code>Float</code>	
homeEndElo - <code>Float</code>	
homePoints - <code>Float</code>	
homePostgameWinProb - <code>Float</code>	
homeStartElo - <code>Float</code>	
homeTeamId - <code>Float</code>	
id - <code>Float</code>	
season - <code>Float</code>	
venueId - <code>Float</code>	
week - <code>Float</code>	
	"homeEndElo": 987.65, "homePoints": 987.65, "homePostgameWinProb": 123.45, "homeStartElo": 987.65, "homeTeamId": 987.65, "id": 987.65, "season": 987.65, "venueId": 123.45, "week": 987.65 }

Types

gameStddevPopFields

aggregate stddevPop on columns

Field Name	Description
attendance - Float	
awayConferenceId - Float	
awayEndElo - Float	
awayPoints - Float	
awayPostgameWinProb - Float	
awayStartElo - Float	
awayTeamId - Float	
excitement - Float	
homeConferenceId - Float	
homeEndElo - Float	

EXAMPLE

```
{  
  "attendance": 987.65,  
  "awayConferenceId": 123.45,  
  "awayEndElo": 987.65,  
  "awayPoints": 123.45,  
  "awayPostgameWinProb": 987.65,  
  "awayStartElo": 123.45,  
  "awayTeamId": 123.45,  
  "excitement": 123.45,  
  "homeConferenceId": 987.65,  
  "homeEndElo": 123.45,  
  "homePoints": 123.45,  
  "homePostgameWinProb": 123.45,  
  "homeStartElo": 123.45,  
  "homeTeamId": 987.65,  
  "id": 123.45,  
  "season": 987.65,  
  "venueId": 123.45,  
  "week": 987.65  
}
```

Field Name	Description
homePoints - Float	
homePostgameWinProb - Float	
homeStartElo - Float	
homeTeamId - Float	
id - Float	
season - Float	
venueId - Float	
week - Float	

Types

gameStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

```
{  
  "attendance": 987.65,  
  "awayConferenceId": 123.45,  
  "awayEndElo": 987.65,
```

Field Name	Description
attendance - Float	
awayConferenceId - Float	
awayEndElo - Float	
awayPoints - Float	
awayPostgameWinProb - Float	
awayStartElo - Float	
awayTeamId - Float	
excitement - Float	
homeConferenceId - Float	
homeEndElo - Float	
homePoints - Float	
homePostgameWinProb - Float	
homeStartElo - Float	
homeTeamId - Float	
id - Float	
	"awayPoints": 123.45, "awayPostgameWinProb": 987.65, "awayStartElo": 987.65, "awayTeamId": 123.45, "excitement": 123.45, "homeConferenceId": 987.65, "homeEndElo": 123.45, "homePoints": 987.65, "homePostgameWinProb": 987.65, "homeStartElo": 987.65, "homeTeamId": 123.45, "id": 987.65, "season": 123.45, "venueId": 123.45, "week": 123.45 }

Field Name	Description
season - Float	
venueId - Float	
week - Float	

Types

gameSumFields

aggregate sum on columns

Field Name	Description
attendance - Int	
awayConferenceId - smallint	
awayEndElo - Int	
awayPoints - smallint	
awayPostgameWinProb - numeric	

EXAMPLE

```
{
  "attendance": 987,
  "awayConferenceId": smallint,
  "awayEndElo": 123,
  "awayPoints": smallint,
  "awayPostgameWinProb": numeric,
  "awayStartElo": 987,
  "awayTeamId": 987,
  "excitement": numeric,
  "homeConferenceId": smallint,
  "homeEndElo": 987,
  "homePoints": smallint,
  "homePostgameWinProb": numeric,
  "homeStartElo": 987,
  "homeTeamId": 123,
  "id": 987,
```

Field Name	Description	
awayStartElo - Int		"season": smallint, "venueId": 123, "week": smallint }
awayTeamId - Int		
excitement - numeric		
homeConferenceId - smallint		
homeEndElo - Int		
homePoints - smallint		
homePostgameWinProb - numeric		
homeStartElo - Int		
homeTeamId - Int		
id - Int		
season - smallint		
venueId - Int		
week - smallint		

Types

gameVarPopFields

aggregate varPop on columns

Field Name	Description
attendance - Float	
awayConferenceId - Float	
awayEndElo - Float	
awayPoints - Float	
awayPostgameWinProb - Float	
awayStartElo - Float	
awayTeamId - Float	
excitement - Float	
homeConferenceId - Float	
homeEndElo - Float	
homePoints - Float	

EXAMPLE

```
{
  "attendance": 123.45,
  "awayConferenceId": 987.65,
  "awayEndElo": 123.45,
  "awayPoints": 123.45,
  "awayPostgameWinProb": 987.65,
  "awayStartElo": 123.45,
  "awayTeamId": 987.65,
  "excitement": 987.65,
  "homeConferenceId": 987.65,
  "homeEndElo": 123.45,
  "homePoints": 987.65,
  "homePostgameWinProb": 987.65,
  "homeStartElo": 987.65,
  "homeTeamId": 123.45,
  "id": 123.45,
  "season": 987.65,
  "venueId": 987.65,
  "week": 987.65
}
```

Field Name	Description
homePostgameWinProb	- Float
homeStartElo	- Float
homeTeamId	- Float
id	- Float
season	- Float
venueId	- Float
week	- Float

Types

gameVarSampFields

aggregate varSamp on columns

EXAMPLE

Field Name	Description
attendance	- Float

```
{  
  "attendance": 123.45,  
  "awayConferenceId": 987.65,  
  "awayEndElo": 987.65,  
  "awayPoints": 987.65,  
  "awayPostgameWinProb": 987.65,
```

Field Name	Description	
awayConferenceId		"awayStartElo": 987.65,
- Float		"awayTeamId": 987.65,
awayEndElo - Float		"excitement": 987.65,
awayPoints - Float		"homeConferenceId": 123.45,
awayPostgameWinProb		"homeEndElo": 123.45,
- Float		"homePoints": 987.65,
awayStartElo - Float		"homePostgameWinProb": 123.45,
awayTeamId - Float		"homeStartElo": 123.45,
excitement - Float		"homeTeamId": 987.65,
homeConferenceId		"id": 123.45,
- Float		"season": 987.65,
homeEndElo - Float		"venueId": 123.45,
homePoints - Float		"week": 123.45
homePostgameWinProb		}
- Float		
homeStartElo - Float		
homeTeamId - Float		
id - Float		
season - Float		

Field Name	Description
venueId - Float	
week - Float	

Types

gameVarianceFields

aggregate variance on columns

EXAMPLE

Field Name	Description
attendance - Float	
awayConferenceId - Float	
awayEndElo - Float	
awayPoints - Float	
awayPostgameWinProb - Float	
awayStartElo - Float	

```
{
  "attendance": 987.65,
  "awayConferenceId": 123.45,
  "awayEndElo": 123.45,
  "awayPoints": 123.45,
  "awayPostgameWinProb": 987.65,
  "awayStartElo": 123.45,
  "awayTeamId": 987.65,
  "excitement": 123.45,
  "homeConferenceId": 987.65,
  "homeEndElo": 123.45,
  "homePoints": 987.65,
  "homePostgameWinProb": 123.45,
  "homeStartElo": 123.45,
  "homeTeamId": 123.45,
  "id": 987.65,
  "season": 123.45,
  "venueId": 987.65,
```

Field Name	Description	
awayTeamId - Float		"week": 123.45
excitement - Float		}
homeConferenceId - Float		
homeEndElo - Float		
homePoints - Float		
homePostgameWinProb - Float		
homeStartElo - Float		
homeTeamId - Float		
id - Float		
season - Float		
venueId - Float		
week - Float		

Types

game_status

EXAMPLE

```
game_status
```

Types

historicalTeam

columns and relationships of
"team_info"

Field Name	Description
abbreviation - String	
active - Boolean	
altColor - String	
altName - String	

EXAMPLE

```
{  
  "abbreviation": "xyz789",  
  "active": true,  
  "altColor": "xyz789",  
  "altName": "abc123",  
  "classification": division,  
  "color": "abc123",  
  "conference": "xyz789",  
  "conferenceAbbreviation": "xyz789",  
  "conferenceId": smallint,  
  "conferenceShortName": "abc123",  
  "countryCode": "xyz789",  
  "displayName": "abc123",  
  "division": "abc123",  
  "endYear": smallint,  
  "id": 987,
```

Field Name	Description	
classification - division		"images": ["xyz789"], "mascot": "abc123", "ncaaName": "abc123", "nickname": "abc123", "school": "xyz789", "shortDisplayName": "xyz789", "startYear": smallint, "twitter": "abc123"
color - String		}
conference - String		
conferenceAbbreviation - String		
conferenceId - smallint		
conferenceShortName - String		
countryCode - String		
displayName - String		
division - String		
endYear - smallint		
id - Int		
images - [String!]		
mascot - String		
ncaaName - String		

Field Name	Description
nickname - <code>String</code>	
school - <code>String</code>	
shortDisplayName - <code>String</code>	
startYear - <code>smallint</code>	
twitter - <code>String</code>	

Types

historicalTeamAggregate

aggregated selection of "team_info"

EXAMPLE

```
{  
  "aggregate": historicalTeamAggreg  
  "nodes": [historicalTeam]  
}
```

Field Name	Description
aggregate - <code>historicalTeamAggregateFields</code>	
nodes - <code>[historicalTeam!]!</code>	

Types

historicalTeamAggregateFields

aggregate fields of "team_info"

Field Name	Description
------------	-------------

avg -

[historicalTeamAvgFields](#)

count - Int!

Arguments

columns -

[\[historicalTeamSelectColumn!\]](#)

distinct - Boolean

max -

[historicalTeamMaxFields](#)

min -

[historicalTeamMinFields](#)

EXAMPLE

```
{  
  "avg": historicalTeamAvgFields,  
  "count": 123,  
  "max": historicalTeamMaxFields,  
  "min": historicalTeamMinFields,  
  "stddev": historicalTeamStddevFie  
  "stddevPop": historicalTeamStddev  
  "stddevSamp": historicalTeamStdde  
  "sum": historicalTeamSumFields,  
  "varPop": historicalTeamVarPopFie  
  "varSamp": historicalTeamVarSampF  
  "variance": historicalTeamVarianc  
}
```

Field Name	Description
stddev -	historicalTeamStddevFields
stddevPop -	historicalTeamStddevPopFields
stddevSamp -	historicalTeamStddevSampFields
sum -	historicalTeamSumFields
varPop -	historicalTeamVarPopFields
varSamp -	historicalTeamVarSampFields
variance -	historicalTeamVarianceFields

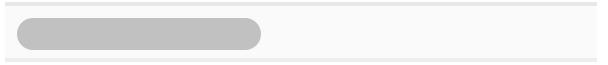
Types

historicalTeamAvgFields

aggregate avg on columns

EXAMPLE

```
{"conferenceId": 987.65, "endYear":
```



Field Name	Description
conferenceId -	
Float	
endYear -	Float
id -	Float
startYear -	Float

Types

historicalTeamBoolExp

Boolean expression to filter rows from the table "team_info". All fields are combined with a logical 'AND'.

EXAMPLE

```
{
  "_and": [historicalTeamBoolExp],
  "_not": historicalTeamBoolExp,
  "_or": [historicalTeamBoolExp],
  "abbreviation": StringComparisonE
  "active": BooleanComparisonExp,
  "altColor": StringComparisonExp,
  "altName": StringComparisonExp,
  "classification": DivisionCompa
  "color": StringComparisonExp,
```

Input Field	Description
_and -	
[historicalTeamBoolExp!]	

Input Field	Description
_not -	
historicalTeamBoolExp	
_or -	
[historicalTeamBoolExp!]	
abbreviation -	
StringComparisonExp	
active -	
BooleanComparisonExp	
altColor -	
StringComparisonExp	
altName -	
StringComparisonExp	
classification -	
DivisionComparisonExp	
color -	
StringComparisonExp	
conference -	
StringComparisonExp	
conferenceAbbreviation	
-	
StringComparisonExp	

```
"conference": StringComparisonExp
"conferenceAbbreviation": StringComparisonExp
"conferenceId": SmallIntComparisonExp
"conferenceShortName": StringComparisonExp
"countryCode": StringComparisonExp
"displayName": StringComparisonExp
"division": StringComparisonExp,
"endYear": SmallIntComparisonExp,
"id": IntComparisonExp,
"images": StringArrayComparisonExp
"mascot": StringComparisonExp,
"ncaaName": StringComparisonExp,
"nickname": StringComparisonExp,
"school": StringComparisonExp,
"shortDisplayName": StringComparisonExp
"startYear": SmallIntComparisonExp
"twitter": StringComparisonExp
}
```

Input Field	Description
conferenceId -	SmallIntComparisonExp
conferenceShortName	-
	StringComparisonExp
countryCode -	StringComparisonExp
displayName -	StringComparisonExp
division -	StringComparisonExp
endYear -	SmallIntComparisonExp
id -	IntComparisonExp
images -	StringArrayComparisonExp
mascot -	StringComparisonExp
ncaaName -	StringComparisonExp

Input Field	Description
nickname -	
	StringComparisonExp
school -	
	StringComparisonExp
shortDisplayName	
-	
	StringComparisonExp
startYear -	
	SmallintComparisonExp
twitter -	
	StringComparisonExp

Types

historicalTeamMaxFields

aggregate max on columns

EXAMPLE

```
{  
  "abbreviation": "xyz789",  
  "altColor": "abc123",  
  "altName": "abc123",  
}
```

Field Name	Description
abbreviation - String	
altColor - String	
altName - String	
classification - division	
color - String	
conference - String	
conferenceAbbreviation - String	
conferenceId - smallint	
conferenceShortName - String	
countryCode - String	
displayName - String	
division - String	
endYear - smallint	

```

"classification": division,
"color": "xyz789",
"conference": "abc123",
"conferenceAbbreviation": "xyz789"
"conferenceId": smallint,
"conferenceShortName": "xyz789",
"countryCode": "abc123",
"displayName": "xyz789",
"division": "abc123",
"endYear": smallint,
"id": 123,
"images": ["xyz789"],
"mascot": "abc123",
"ncaaName": "xyz789",
"nickname": "abc123",
"school": "xyz789",
"shortDisplayName": "abc123",
"startYear": smallint,
"twitter": "abc123"
}

```

Field Name	Description
id - <code>Int</code>	
images - <code>[String!]</code>	
mascot - <code>String</code>	
ncaaName - <code>String</code>	
nickname - <code>String</code>	
school - <code>String</code>	
shortDisplayName - <code>String</code>	
startYear - <code>smallint</code>	
twitter - <code>String</code>	

Types

historicalTeamMinFields

aggregate min on columns

EXAMPLE

```
{  
  "abbreviation": "xyz789",
```

Field Name	Description
abbreviation - String	
altColor - String	
altName - String	
classification - division	
color - String	
conference - String	
conferenceAbbreviation - String	
conferenceId - smallint	
conferenceShortName - String	
countryCode - String	
displayName - String	
division - String	
endYear - smallint	
	"altColor": "xyz789", "altName": "abc123", "classification": division, "color": "xyz789", "conference": "abc123", "conferenceAbbreviation": "abc123" "conferenceId": smallint, "conferenceShortName": "abc123", "countryCode": "abc123", "displayName": "xyz789", "division": "xyz789", "endYear": smallint, "id": 123, "images": ["abc123"], "mascot": "abc123", "ncaaName": "abc123", "nickname": "xyz789", "school": "xyz789", "shortDisplayName": "abc123", "startYear": smallint, "twitter": "xyz789" }

Field Name	Description
id - <code>Int</code>	
images - <code>[String!]</code>	
mascot - <code>String</code>	
ncaaName - <code>String</code>	
nickname - <code>String</code>	
school - <code>String</code>	
shortDisplayName - <code>String</code>	
startYear - <code>smallint</code>	
twitter - <code>String</code>	

Types

historicalTeamOrderBy

Ordering options when selecting data from "team_info".

EXAMPLE

```
{  
  "abbreviation": "ASC",
```

Input Field	Description
abbreviation - OrderBy	"active": "ASC", "altColor": "ASC", "altName": "ASC", "classification": "ASC", "color": "ASC", "conference": "ASC", "conferenceAbbreviation": "ASC", "conferenceId": "ASC", "conferenceShortName": "ASC", "countryCode": "ASC", "displayName": "ASC", "division": "ASC", "endYear": "ASC", "id": "ASC", "images": "ASC", "mascot": "ASC", "ncaaName": "ASC", "nickname": "ASC", "school": "ASC", "shortDisplayName": "ASC", "startYear": "ASC", "twitter": "ASC"
active - OrderBy	}
altColor - OrderBy	
altName - OrderBy	
classification - OrderBy	
color - OrderBy	
conference - OrderBy	
conferenceAbbreviation - OrderBy	
conferenceId - OrderBy	
conferenceShortName - OrderBy	
countryCode - OrderBy	
displayName - OrderBy	
division - OrderBy	

Input Field	Description
endYear - OrderBy	
id - OrderBy	
images - OrderBy	
mascot - OrderBy	
ncaaName - OrderBy	
nickname - OrderBy	
school - OrderBy	
shortDisplayName - OrderBy	
startYear - OrderBy	
twitter - OrderBy	

Types

historicalTeamSelectColumn

select columns of table "team_info"

[EXAMPLE](#)

Enum Value	Description	
abbreviation	column name	"abbreviation"
active	column name	
altColor	column name	
altName	column name	
classification	column name	
color	column name	
conference	column name	
conferenceAbbreviation	column name	
conferenceId	column name	
conferenceShortName	column name	
countryCode	column name	
displayName	column name	
division	column name	
endYear	column name	
id	column name	
images	column name	
mascot	column name	
ncaaName	column name	
nickname	column name	

Enum Value	Description
school	column name
shortDisplayName	column name
startYear	column name
twitter	column name

Types

historicalTeamStddevFields

aggregate stddev on columns

EXAMPLE

```
{"conferenceId": 123.45, "endYear":
```

Field Name	Description
conferenceId -	
Float	
endYear -	Float
id -	Float
startYear -	Float

Types

historicalTeamStddevPopFields

aggregate stddevPop on columns

EXAMPLE

```
{"conferenceId": 987.65, "endYear":
```

Field Name	Description
conferenceId -	
Float	
endYear -	Float
id -	Float
startYear -	Float



Types

historicalTeamStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
conferenceId -	
Float	
endYear -	Float
id -	Float
startYear -	Float

```
{"conferenceId": 123.45, "endYear":
```

Types

historicalTeamSumFields

aggregate sum on columns

EXAMPLE

Field Name	Description
conferenceId -	
smallint	
endYear -	smallint
id -	Int
startYear -	
smallint	

```
{  
    "conferenceId": smallint,  
    "endYear": smallint,  
    "id": 987,  
    "startYear": smallint  
}
```

Types

historicalTeamVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
conferenceId -	
Float	
endYear -	Float
Float	
id -	Float
Float	
startYear -	Float
Float	

{ "conferenceId": 987.65, "endYear":



Types

historicalTeamVarSampFields

aggregate varSamp on columns

EXAMPLE

```
{"conferenceId": 987.65, "endYear":
```

Field Name	Description
conferenceId -	
Float	
endYear -	Float
Float	
id -	Float
Float	
startYear -	Float
Float	

Types

historicalTeamVarianceFields

aggregate variance on columns

EXAMPLE

```
{"conferenceId": 123.45, "endYear":
```

Field Name	Description
conferenceId -	
Float	
endYear -	Float
Float	
id -	Float
Float	

Field Name	Description
startYear - Float	

Types

home_away

EXAMPLE

[home_away](#)

Types

media_type

EXAMPLE

[media_type](#)

Types

numeric

EXAMPLE

`numeric`

Types

player_adjusted_metric_type

EXAMPLE

`player_adjusted_metric_type`

Types

predictedPoints

columns and relationships of "ppa"

Field Name	Description
distance -	
smallint!	
down - smallint!	
predictedPoints -	
numeric!	
yardLine -	
smallint!	

EXAMPLE

```
{
  "distance": smallint,
  "down": smallint,
  "predictedPoints": numeric,
  "yardLine": smallint
}
```

Types

predictedPointsAggregate

aggregated selection of "ppa"

Field Name	Description
aggregate -	
predictedPointsAggregateFields	

EXAMPLE

```
{
  "aggregate": predictedPointsAggre
  "nodes": [predictedPoints]
}
```

Field Name	Description
nodes -	[predictedPoints!]!

Types

predictedPointsAggregateFields

aggregate fields of "ppa"

EXAMPLE

Field Name	Description
avg -	predictedPointsAvgFields
count - Int!	
Arguments	
columns -	[predictedPointsSelectColumn!]
distinct - Boolean	

```
{
  "avg": predictedPointsAvgFields,
  "count": 123,
  "max": predictedPointsMaxFields,
  "min": predictedPointsMinFields,
  "stddev": predictedPointsStddevFi
  "stddevPop": predictedPointsStdde
  "stddevSamp": predictedPointsStdd
  "sum": predictedPointsSumFields,
  "varPop": predictedPointsVarPopFi
  "varSamp": predictedPointsVarSamp
  "variance": predictedPointsVarian
}
```



Field Name	Description
max -	predictedPointsMaxFields
min -	predictedPointsMinFields
stddev -	predictedPointsStddevFields
stddevPop -	predictedPointsStddevPopFields
stddevSamp -	predictedPointsStddevSampFields
sum -	predictedPointsSumFields
varPop -	predictedPointsVarPopFields
varSamp -	predictedPointsVarSampFields
variance -	predictedPointsVarianceFields

Types

predictedPointsAvgFields

aggregate avg on columns

Field Name	Description
distance - Float	
down - Float	
predictedPoints - Float	
yardLine - Float	

EXAMPLE

```
{  
  "distance": 987.65,  
  "down": 987.65,  
  "predictedPoints": 123.45,  
  "yardLine": 123.45  
}
```

Types

predictedPointsBoolExp

Boolean expression to filter rows from the table "ppa". All fields are combined with a logical 'AND'.

EXAMPLE

```
{  
  "_and": [predictedPointsBoolExp],  
  "_not": predictedPointsBoolExp,  
  "_or": [predictedPointsBoolExp],  
  "distance": SmallIntComparisonExp
```

Input Field	Description
_and -	
	[predictedPointsBoolExp!]
_not -	
	predictedPointsBoolExp
_or -	
	[predictedPointsBoolExp!]
distance -	
	SmallintComparisonExp
down -	
	SmallintComparisonExp
predictedPoints -	
	NumericComparisonExp
yardLine -	
	SmallintComparisonExp

Types

predictedPointsMaxFields

aggregate max on columns

Field Name	Description
distance -	
smallint	
down - smallint	
predictedPoints -	
numeric	
yardLine -	
smallint	

EXAMPLE

```
{
  "distance": smallint,
  "down": smallint,
  "predictedPoints": numeric,
  "yardLine": smallint
}
```

Types

predictedPointsMinFields

aggregate min on columns

EXAMPLE

Field Name	Description
distance -	
smallint	

```
{
  "distance": smallint,
  "down": smallint,
  "predictedPoints": numeric,
  "yardLine": smallint
}
```

Field Name	Description
down - smallint	
predictedPoints - numeric	
yardLine - smallint	

Types

predictedPointsOrderBy

Ordering options when selecting data from "ppa".

EXAMPLE

```
{"distance": "ASC", "down": "ASC",
```

Input Field	Description
distance - OrderBy	
down - OrderBy	
predictedPoints - OrderBy	
yardLine - OrderBy	



Types

predictedPointsSelectColumn

select columns of table "ppa"

EXAMPLE

Enum Value	Description	
distance	column name	"distance"
down	column name	
predictedPoints	column name	
yardLine	column name	

Types

predictedPointsStddevFields

aggregate stddev on columns

EXAMPLE

Field Name	Description
distance - Float	
down - Float	
predictedPoints - Float	
yardLine - Float	

```
{
  "distance": 123.45,
  "down": 123.45,
  "predictedPoints": 123.45,
  "yardLine": 987.65
}
```

Types

predictedPointsStddevPopFields

aggregate stddevPop on columns

EXAMPLE

Field Name	Description
distance - Float	
down - Float	
predictedPoints - Float	
yardLine - Float	

```
{
  "distance": 987.65,
  "down": 987.65,
  "predictedPoints": 987.65,
  "yardLine": 123.45
}
```

Types

predictedPointsStddevSampFields

aggregate stddevSamp on columns

EXAMPLE

Field Name	Description
distance - Float	
down - Float	
predictedPoints - Float	
yardLine - Float	

```
{  
  "distance": 987.65,  
  "down": 123.45,  
  "predictedPoints": 123.45,  
  "yardLine": 987.65  
}
```

Types

predictedPointsSumFields

aggregate sum on columns

Field Name	Description
distance -	
smallint	
down -	smallint
predictedPoints -	
numeric	
yardLine -	
smallint	

EXAMPLE

```
{
  "distance": smallint,
  "down": smallint,
  "predictedPoints": numeric,
  "yardLine": smallint
}
```

Types

predictedPointsVarPopFields

aggregate varPop on columns

EXAMPLE

Field Name	Description
distance -	Float
down -	Float

```
{
  "distance": 123.45,
  "down": 123.45,
  "predictedPoints": 123.45,
  "yardLine": 987.65
}
```

Field Name	Description
predictedPoints -	
Float	
yardLine -	Float

Types

predictedPointsVarSampFields

aggregate varSamp on columns

EXAMPLE

```
{  
    "distance": 987.65,  
    "down": 123.45,  
    "predictedPoints": 123.45,  
    "yardLine": 987.65  
}
```

Field Name	Description
distance -	Float
down -	Float
predictedPoints -	
Float	
yardLine -	Float

Types

predictedPointsVarianceFields

aggregate variance on columns

Field Name	Description
distance - Float	
down - Float	
predictedPoints - Float	
yardLine - Float	

EXAMPLE

```
{  
  "distance": 123.45,  
  "down": 987.65,  
  "predictedPoints": 987.65,  
  "yardLine": 123.45  
}
```

Types

ratings

columns and relationships of
"rating_systems"

EXAMPLE

```
{  
  "conference": "abc123",  
  "rating_system": "PFF",  
  "team": "Harvard",  
  "year": 2021  
}
```

Field Name	Description
conference -	
String	
conferenceId -	
smallint	
elo -	<code>Int</code>
fpi -	<code>numeric</code>
fpiAvgWinProbabilityRank	
-	<code>smallint</code>
fpiDefensiveEfficiency	
-	<code>numeric</code>
fpiGameControlRank	
-	<code>smallint</code>
fpiOffensiveEfficiency	
-	<code>numeric</code>
fpiOverallEfficiency	
-	<code>numeric</code>
fpiRemainingSosRank	
-	<code>smallint</code>
fpiResumeRank -	
smallint	
fpiSosRank -	
smallint	

```

"conferenceId": smallint,
"elo": 987,
"fpi": numeric,
"fpiAvgWinProbabilityRank": smallint,
"fpiDefensiveEfficiency": numeric,
"fpiGameControlRank": smallint,
"fpiOffensiveEfficiency": numeric,
"fpiOverallEfficiency": numeric,
"fpiRemainingSosRank": smallint,
"fpiResumeRank": smallint,
"fpiSosRank": smallint,
"fpiSpecialTeamsEfficiency": numeric,
"fpiStrengthOfRecordRank": smallint,
"spDefense": numeric,
"spOffense": numeric,
"spOverall": numeric,
"spSpecialTeams": numeric,
"srs": numeric,
"team": "abc123",
"teamId": 987,
"year": smallint
}

```

Field Name	Description
fpiSpecialTeamsEfficiency	- numeric
fpiStrengthOfRecordRank	- smallint
spDefense	- numeric
spOffense	- numeric
spOverall	- numeric
spSpecialTeams	- numeric
srs	- numeric
team	- String
teamId	- Int
year	- smallint

Types

ratingsBoolExp

Boolean expression to filter rows from the table "rating_systems". All fields are combined with a logical 'AND'.

Input Field	Description
_and -	[ratingsBoolExp!]
_not -	ratingsBoolExp
_or -	[ratingsBoolExp!]
conference -	StringComparisonExp
conferenceId -	SmallintComparisonExp
elo -	IntComparisonExp
fpi -	NumericComparisonExp
fpiAvgWinProbabilityRank	-

EXAMPLE

```
{
  "_and": [ratingsBoolExp],
  "_not": ratingsBoolExp,
  "_or": [ratingsBoolExp],
  "conference": StringComparisonExp,
  "conferenceId": SmallintComparisonExp,
  "elo": IntComparisonExp,
  "fpi": NumericComparisonExp,
  "fpiAvgWinProbabilityRank": SmallintComparisonExp,
  "fpiDefensiveEfficiency": NumericComparisonExp,
  "fpiGameControlRank": SmallintComparisonExp,
  "fpiOffensiveEfficiency": NumericComparisonExp,
  "fpiOverallEfficiency": NumericComparisonExp,
  "fpiRemainingSosRank": SmallintComparisonExp,
  "fpiResumeRank": SmallintComparisonExp,
  "fpiSosRank": SmallintComparisonExp,
  "fpiSpecialTeamsEfficiency": NumericComparisonExp,
  "fpiStrengthOfRecordRank": SmallintComparisonExp,
  "spDefense": NumericComparisonExp,
  "spOffense": NumericComparisonExp,
  "spOverall": NumericComparisonExp,
  "spSpecialTeams": NumericComparisonExp,
  "srs": NumericComparisonExp,
  "team": StringComparisonExp,
  "teamId": IntComparisonExp,
  "year": SmallintComparisonExp
}
```

Input Field	Description
SmallintComparisonExp	
fpiDefensiveEfficiency	-
	NumericComparisonExp
fpiGameControlRank	-
	SmallintComparisonExp
fpiOffensiveEfficiency	-
	NumericComparisonExp
fpiOverallEfficiency	-
	NumericComparisonExp
fpiRemainingSosRank	-
	SmallintComparisonExp
fpiResumeRank	-
	SmallintComparisonExp
fpiSosRank	-
	SmallintComparisonExp
fpiSpecialTeamsEfficiency	-
	NumericComparisonExp

Input Field	Description
fpiStrengthOfRecordRank	-
	SmallIntComparisonExp
spDefense	-
	NumericComparisonExp
spOffense	-
	NumericComparisonExp
spOverall	-
	NumericComparisonExp
spSpecialTeams	-
	NumericComparisonExp
srs	-
	NumericComparisonExp
team	-
	StringComparisonExp
teamId	-
	IntComparisonExp
year	-
	SmallIntComparisonExp

Types

ratingsOrderBy

Ordering options when selecting data from "rating_systems".

Input Field	Description
conference - OrderBy	
conferenceId - OrderBy	
elo - OrderBy	
fpi - OrderBy	
fpiAvgWinProbabilityRank - OrderBy	
fpiDefensiveEfficiency - OrderBy	
fpiGameControlRank - OrderBy	
fpiOffensiveEfficiency - OrderBy	
fpiOverallEfficiency - OrderBy	

EXAMPLE

```
{  
  "conference": "ASC",  
  "conferenceId": "ASC",  
  "elo": "ASC",  
  "fpi": "ASC",  
  "fpiAvgWinProbabilityRank": "ASC"  
  "fpiDefensiveEfficiency": "ASC",  
  "fpiGameControlRank": "ASC",  
  "fpiOffensiveEfficiency": "ASC",  
  "fpiOverallEfficiency": "ASC",  
  "fpiRemainingSosRank": "ASC",  
  "fpiResumeRank": "ASC",  
  "fpiSosRank": "ASC",  
  "fpiSpecialTeamsEfficiency": "ASC"  
  "fpiStrengthOfRecordRank": "ASC",  
  "spDefense": "ASC",  
  "spOffense": "ASC",  
  "spOverall": "ASC",  
  "spSpecialTeams": "ASC",  
  "srs": "ASC",  
  "team": "ASC",  
  "teamId": "ASC",  
  "year": "ASC"  
}
```

Input Field	Description
fpiRemainingSosRank	
- OrderBy	
fpiResumeRank -	
OrderBy	
fpiSosRank -	
OrderBy	
fpiSpecialTeamsEfficiency	
- OrderBy	
fpiStrengthOfRecordRank	
- OrderBy	
spDefense -	
OrderBy	
spOffense -	
OrderBy	
spOverall -	
OrderBy	
spSpecialTeams -	
OrderBy	
srs - OrderBy	
team - OrderBy	
teamId - OrderBy	

Input Field	Description
year - OrderBy	

Types

ratingsSelectColumn

select columns of table
"rating_systems"

EXAMPLE

"conference"

Enum Value	Description
conference	column name
conferenceId	column name
elo	column name
fpi	column name
fpiAvgWinProbability	column name
fpiDefensiveEfficiency	column name
fpiGameControlRank	column name
fpiOffensiveEfficiency	column name

Enum Value	Description
fpiOverallEfficiency	column name
fpiRemainingSosRank	column name
fpiResumeRank	column name
fpiSosRank	column name
fpiSpecialTeamsEfficiency	column name
fpiStrengthOfRecord	column name
spDefense	column name
spOffense	column name
spOverall	column name
spSpecialTeams	column name
srs	column name
team	column name
teamId	column name
year	column name

Types

recruitAggregateBoolExpCount

Input Field	Description	EXAMPLE
arguments -		
	[RecruitSelectColumn!]	
distinct -	Boolean	
filter -		
	RecruitBoolExp	
predicate -		
	IntComparisonExp!	

Types

recruit_type

EXAMPLE

recruit_type

Types

season_type

EXAMPLE`season_type`

Types

smallint

EXAMPLE`smallint`

Types

timestamp

EXAMPLE`timestamp`

Types

timestamptz

EXAMPLE`timestamptz`

[Documentation by Anvil SpectaQL](#)