Dani Schnider

In Search of Use Cases for Data Use Case Domains

Callista

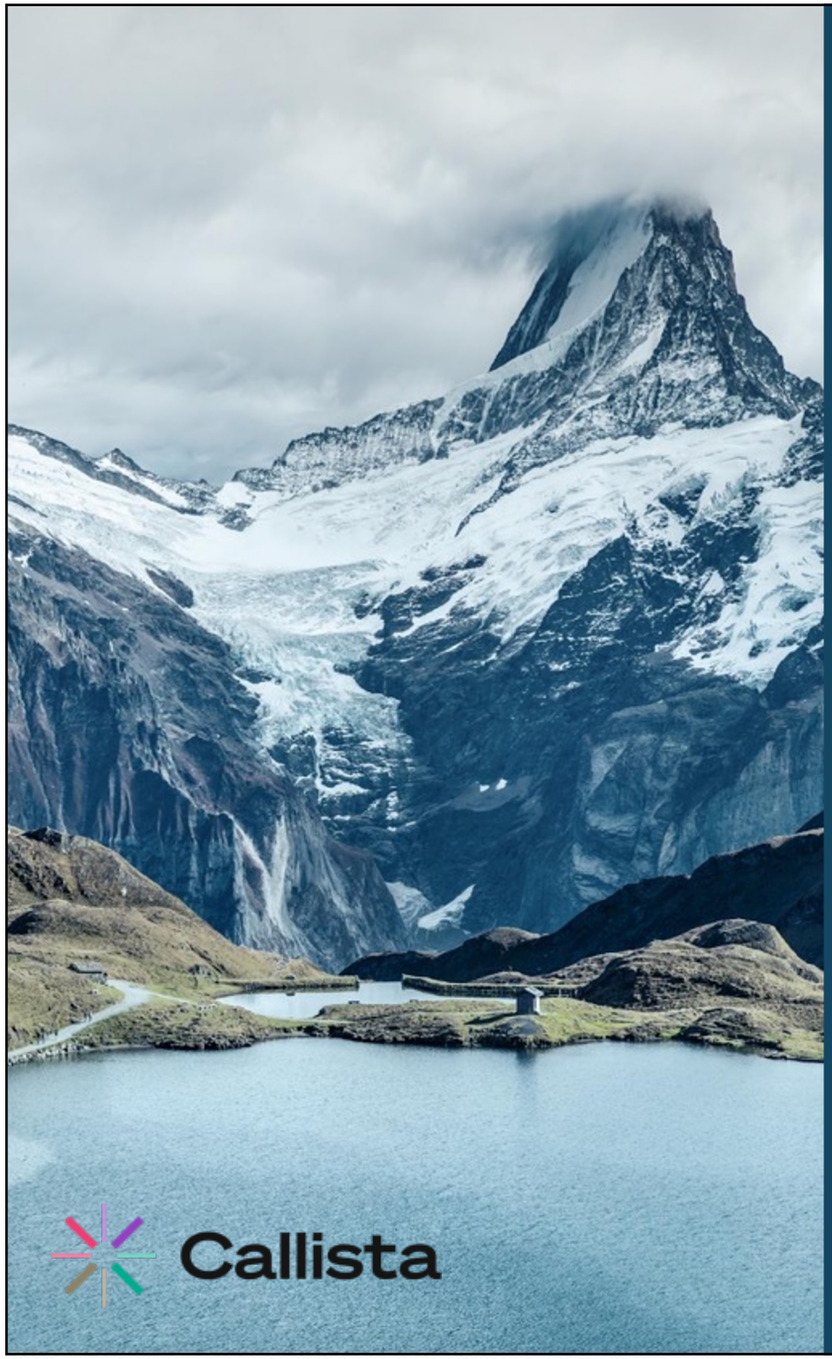
About me

Dani Schnider

- Working for Callista
- Oracle ACE Director
- Member of Symposium 42
- Hobby: Craft Beer Brewing

@danischnider.bsky.social danischnider.wordpress.com www.linkedin.com/in/danischnider/

Callista



What are Data Use Case Domains?

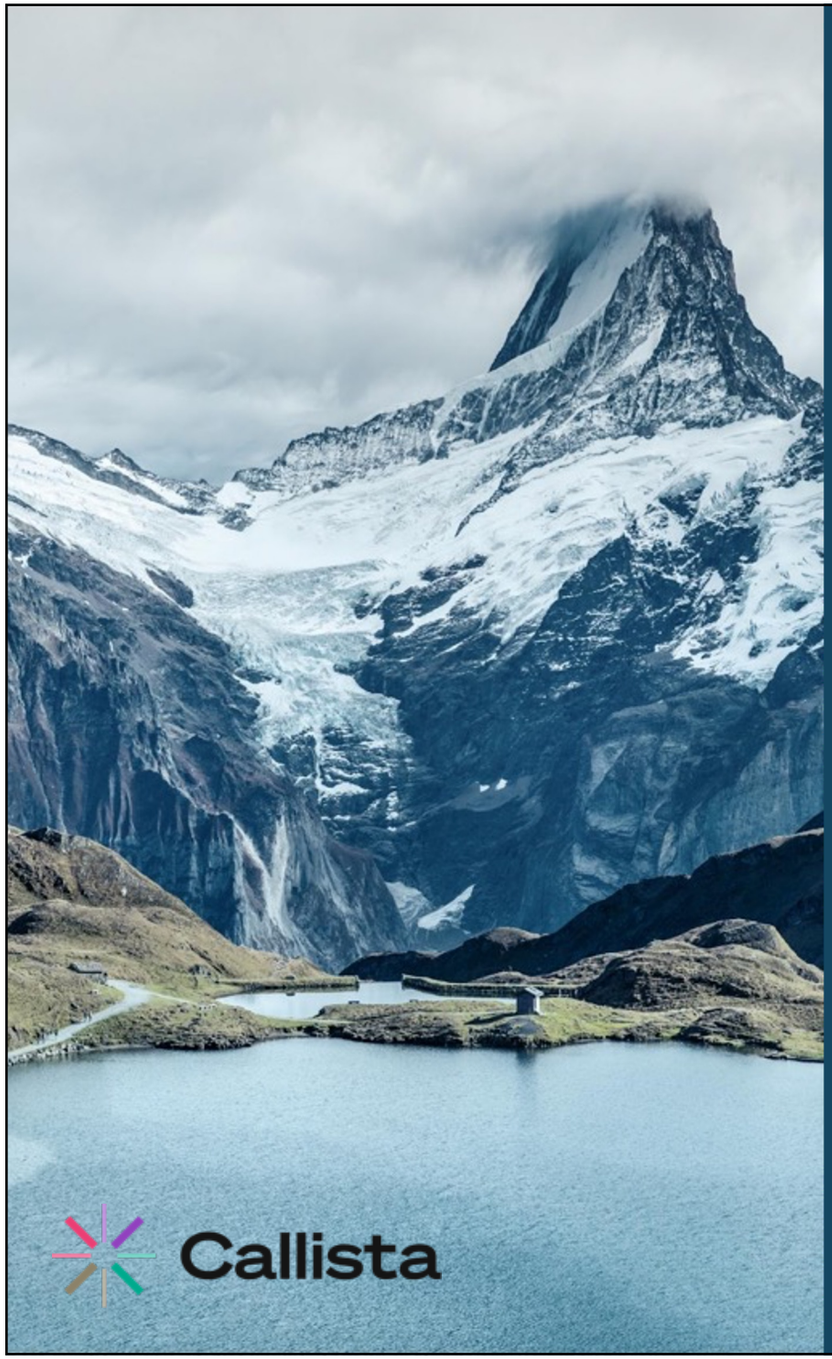
ChatGPT

What can I help with?

Message ChatGPT

Search Reason

Create image Make a plan Get advice Analyze data Code More



What are Data Use Case Domains in Oracle Database 23ai?

Data Use Case Domains in Oracle 23ai

- Data dictionary objects defining common value properties:
  - Column properties
  - Data types
  - Enumerations
  - Check constraints
  - Display options
  - Sort order
  - Annotations (extended metadata / comments)

SQL Domains vs. Data Use Case Domains

- SQL Domains
  - Defined in SQL standard (SQL-92, SQL:1999)
  - Available in PostgreSQL, IBM Db2 and other RDBMS
  - …but not in Oracle (yet)

- Data Use Case Domains
  - Extension of SQL Domains
  - Single Column Domains, Multi Column Domain, Flexible Domains
  - Introduced in Oracle Database 23ai

SQL:1999 Standard

```
CREATE DOMAIN <Domain name> [ AS ] <data type>
     [ DEFAULT default value ]
     [ <Domain Constraint> list ]
     [ COLLATE <Collation name> ]

<Domain constraint> list::=
   <Domain Constraint> [ <Domain Constraint>... ]

<Domain constraint> ::=
      [ CONSTRAINT <Constraint name> ]
      Constraint_type
      [ <constraint attributes> ]
```

https://sql-99.readthedocs.io/en/latest/chapters/19.html (SQL-99 Complete, Really)

PostgreSQL 17.3

```
CREATE DOMAIN name [ AS ] data_type
    [ COLLATE collation ]
    [ DEFAULT expression ]
    [ domain_constraint [ ... ] ]

domain_constraint::=
    [ CONSTRAINT constraint_name ]
    { NOT NULL | NULL | CHECK (expression) }
```

https://www.postgresql.org/docs/current/sql-createdomain.html

Compatibility to SQL:1999

"The command CREATE DOMAIN conforms to the SQL standard.

The syntax NOT NULL in this command is a PostgreSQL extension."

Oracle 23ai

```
CREATE [ USECASE ] DOMAIN [IF NOT EXISTS ][ schema .] domain_name
AS { datatype   |   ENUM ( enum_list ) }
    [ STRICT ] [column_properties_clause]
    [ DISPLAY display_expression ]
    [ ORDER order_expression ]
    [ annotations_clause ]

column_properties_clause::=
   [ default_clause ]
   [NULL | NOT NULL ]
   [ constraint_clause ]
   [VALIDATE USING schema_constant_text ]
   [COLLATE column_collation_name ]
```

https://docs.oracle.com/en/database/oracle/oracle-database/23/sqlrf/create-domain.html

Callista

Single Column Domains

Compatibility to SQL:1999

"Note: Oracle's domain implementation provides a wider range of functions than the one specified in the SQL standard"

(Blog post "Less coding using domains" from Ulrike Schwinn https://blogs.oracle.com/coretec/post/less-coding-with-sql-domains-in-23c)

Oracle 23ai

Multi Column Domains

```
CREATE [ USECASE ] DOMAIN [ IF NOT EXISTS ][ schema .] domain_name
AS ( domain_column AS datatype [ STRICT ] [ column_properties_clause ]
   [, domain_column AS datatype [ STRICT ] [ column_properties_clause ] )
   [DISPLAY display_expression ]
   [ORDER order_expression ]
   [annotations_clause ]

column_properties_clause::=
  [ default_clause ]
  [NULL | NOT NULL ]
  [ constraint_clause ]
  [VALIDATE USING schema_constant_text ]
  [COLLATE column_collation_name ]
```

https://docs.oracle.com/en/database/oracle/oracle-database/23/sqlrf/create-domain.html

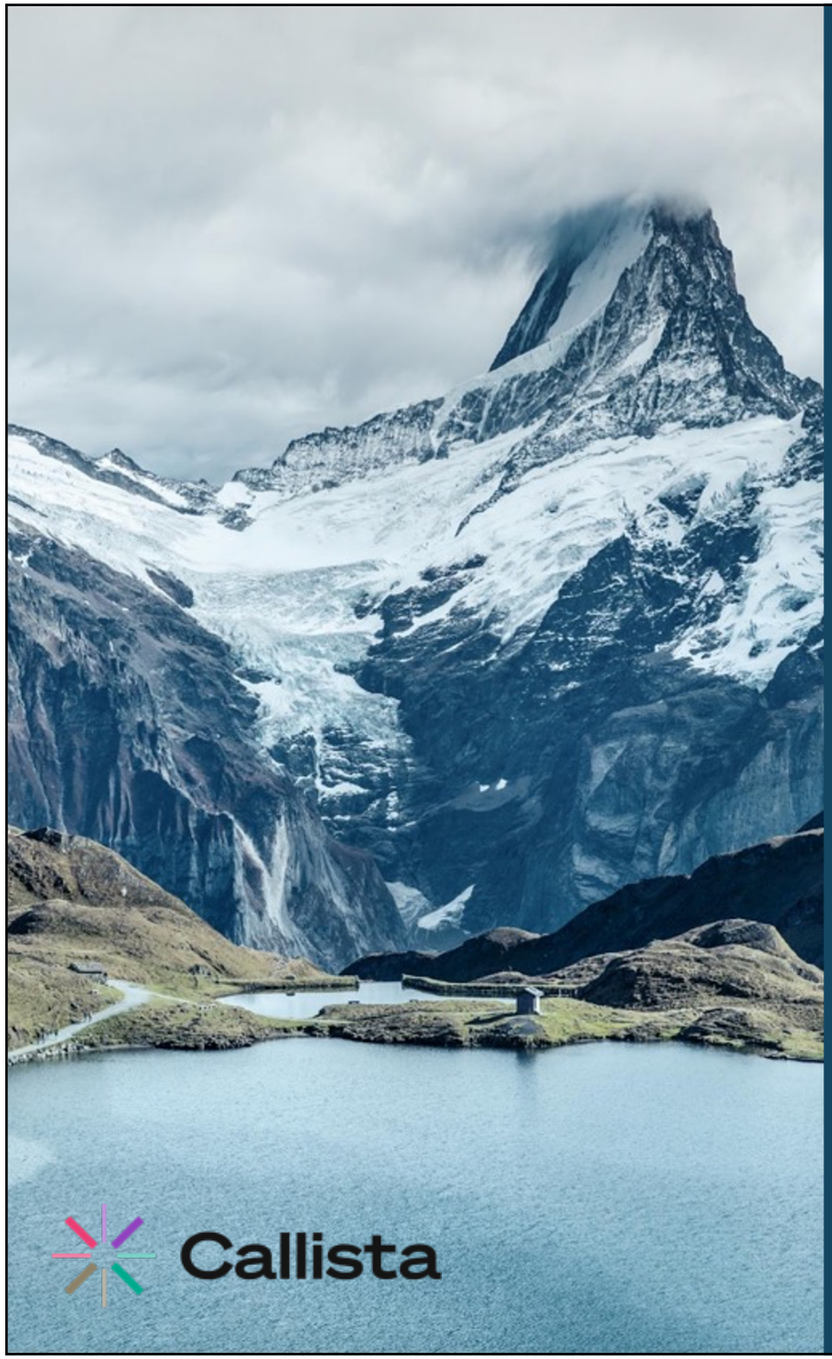
Oracle 23ai

Flexible Domains

```
CREATE [ USECASE ] FLEXIBLE DOMAIN [IF NOT EXISTS ][ schema .]domain_name
    ( domain_column [ , domain_column... ] )
    CHOOSE DOMAIN USING ( domain_discriminant_column datatype)
    [ , domain_discriminant_column datatype...] )
FROM
  { DECODE (expr , (search_expr , result_expr)
  [, search_expr , result_expr ]...[ , default ] ) | case_expression
  }
result_expr::=
[ schema .]domain_name ( (domain_column) [, (domain_column)...] )
default_clause::=
DEFAULT [ ON NULL [FOR ( INSERT { ONLY | AND UPDATE } ) ] ]
default_expression
```

https://docs.oracle.com/en/database/oracle/oracle-database/23/sqlrf/create-domain.html

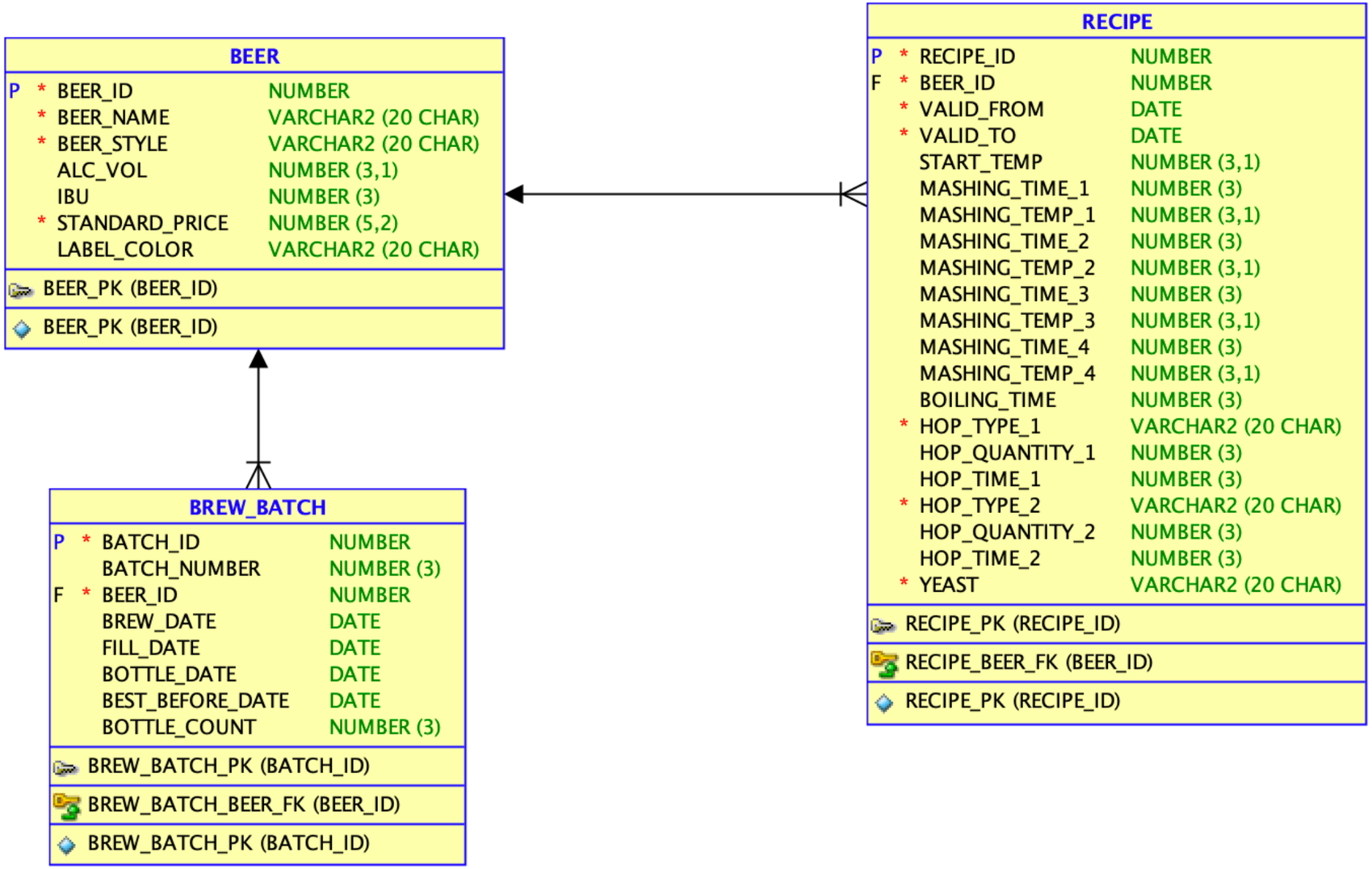
Callista



What are Use Cases for Data Use Case Domains?

My Use Case for this Presentation

Demo Example



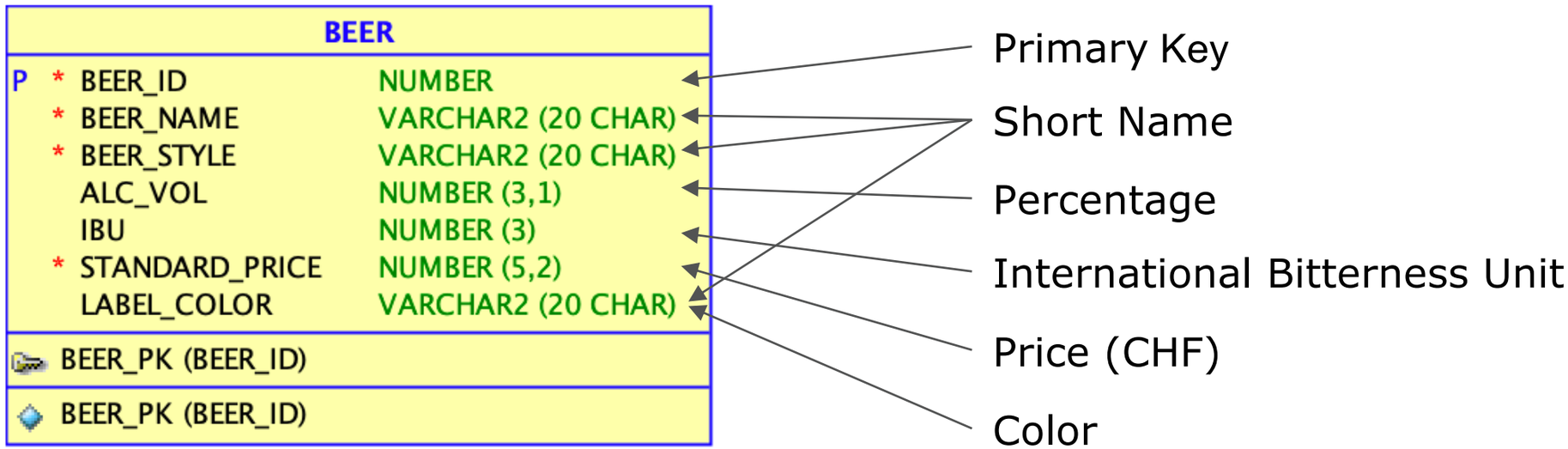
Callista

Table BEER

BEER		
P *	BEER_ID	NUMBER
*	BEER_NAME	VARCHAR2 (20 CHAR)
*	BEER_STYLE	VARCHAR2 (20 CHAR)
	ALC_VOL	NUMBER (3,1)
	IBU	NUMBER (3)
*	STANDARD_PRICE	NUMBER (5,2)
	LABEL_COLOR	VARCHAR2 (20 CHAR)
BEER_PK (BEER_ID)		
BEER_PK (BEER_ID)		

Callista

Table BEER



Callista

Live Demo


```
SQL> desc beer

Name                 Null?       Type
-------------------- ----------- ----------------------------------------
BEER_ID              NOT NULL    NUMBER DOMAIN PRIMARY_KEY
BEER_NAME            NOT NULL    VARCHAR2(20 CHAR) DOMAIN SHORT_NAME
BEER_STYLE           NOT NULL    VARCHAR2(20 CHAR) DOMAIN SHORT_NAME
ALC_VOL                          NUMBER(3,1) DOMAIN PERCENTAGE
IBU                              NUMBER(3) DOMAIN INTERNATIONAL_BITTERNESS_UNIT
STANDARD_PRICE       NOT NULL    NUMBER(5,2) DOMAIN PRICE
LABEL_COLOR                      VARCHAR2(20 CHAR) DOMAIN COLOR
```

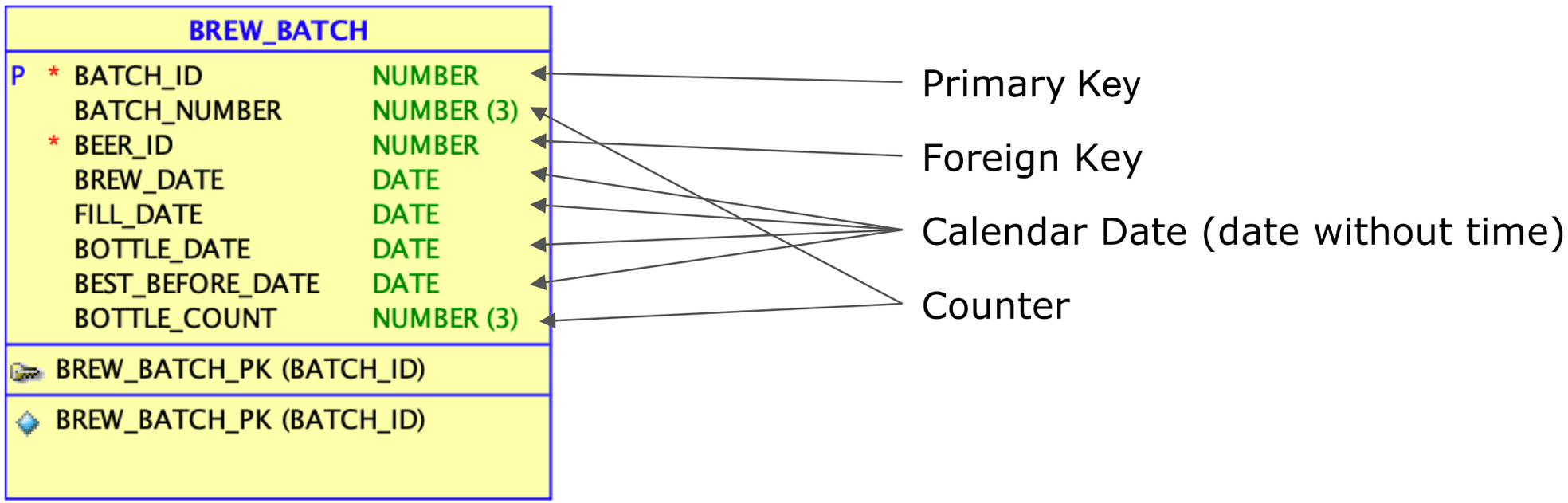
Callista

Table BREW_BATCH

BREW_BATCH		
P *	BATCH_ID	NUMBER
	BATCH_NUMBER	NUMBER (3)
*	BEER_ID	NUMBER
	BREW_DATE	DATE
	FILL_DATE	DATE
	BOTTLE_DATE	DATE
	BEST_BEFORE_DATE	DATE
	BOTTLE_COUNT	NUMBER (3)
BREW_BATCH_PK (BATCH_ID)		
BREW_BATCH_PK (BATCH_ID)		

Callista

Table BREW_BATCH



Callista

Live Demo


```
SQL> desc brew_batch

Name                  Null?       Type
_____________________ ___________ ____________________________
BATCH_ID              NOT NULL    NUMBER DOMAIN PRIMARY_KEY
BATCH_NUMBER                      NUMBER(3) DOMAIN COUNTER
BEER_ID               NOT NULL    NUMBER DOMAIN FOREIGN_KEY
BREW_DATE                         DATE DOMAIN CALENDAR_DATE
FILL_DATE                         DATE DOMAIN CALENDAR_DATE
BOTTLE_DATE                       DATE DOMAIN CALENDAR_DATE
BEST_BEFORE_DATE                  DATE DOMAIN CALENDAR_DATE
BOTTLE_COUNT                      NUMBER(3) DOMAIN COUNTER
```

Callista

Table RECIPE

RECIPE		
P *	RECIPE_ID	NUMBER
*	BEER_ID	NUMBER
*	VALID_FROM	DATE
*	VALID_TO	DATE
	START_TEMP	NUMBER (3,1)
	MASHING_TIME_1	NUMBER (3)
	MASHING_TEMP_1	NUMBER (3,1)
	MASHING_TIME_2	NUMBER (3)
	MASHING_TEMP_2	NUMBER (3,1)
	MASHING_TIME_3	NUMBER (3)
	MASHING_TEMP_3	NUMBER (3,1)
	MASHING_TIME_4	NUMBER (3)
	MASHING_TEMP_4	NUMBER (3,1)
	BOILING_TIME	NUMBER (3)
*	HOP_TYPE_1	VARCHAR2 (20 CHAR)
	HOP_QUANTITY_1	NUMBER (3)
	HOP_TIME_1	NUMBER (3)
*	HOP_TYPE_2	VARCHAR2 (20 CHAR)
	HOP_QUANTITY_2	NUMBER (3)
	HOP_TIME_2	NUMBER (3)
*	YEAST	VARCHAR2 (20 CHAR)
RECIPE_PK (RECIPE_ID)		
RECIPE_PK (RECIPE_ID)		

Callista

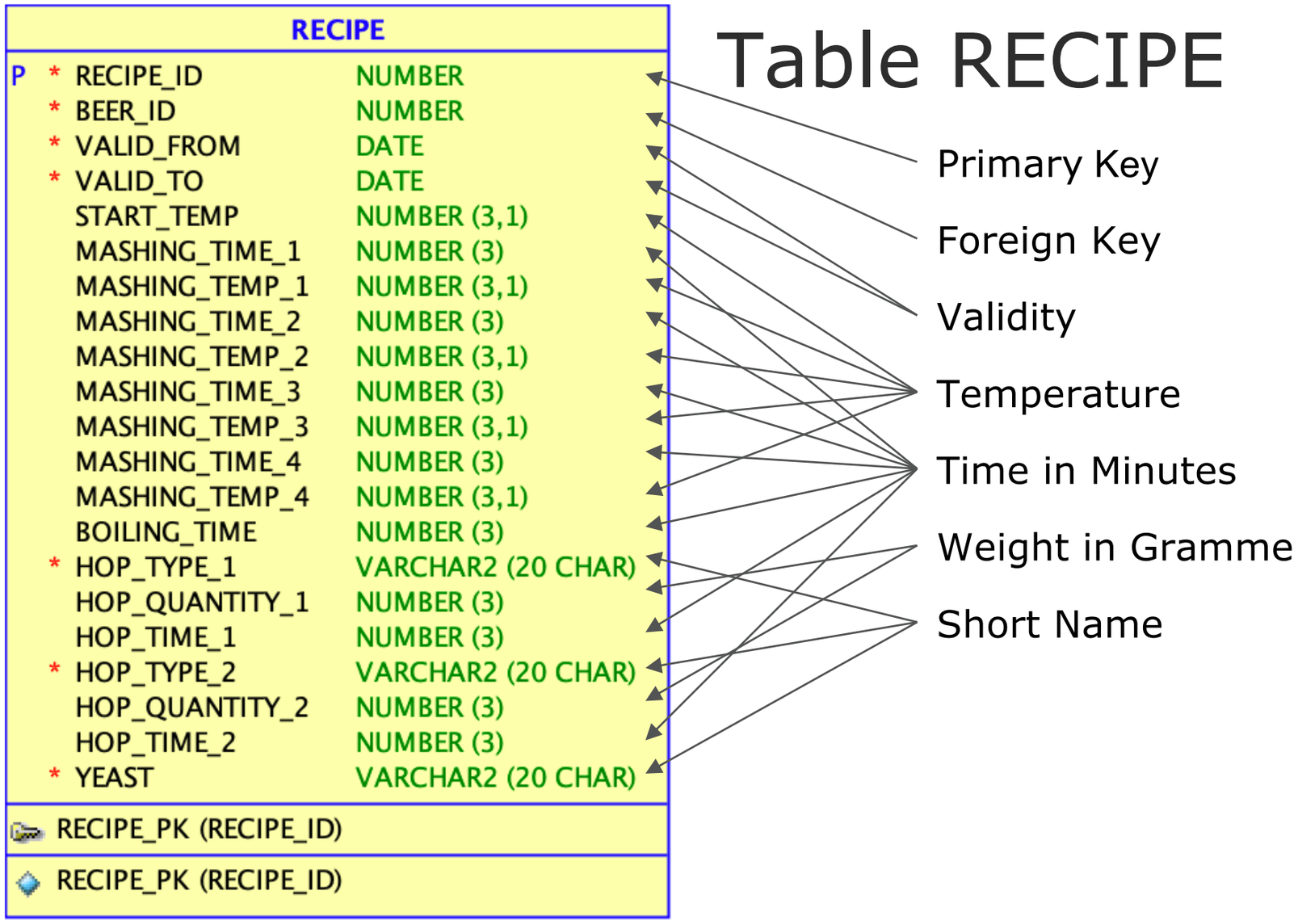


Table RECIPE

Primary Key

Foreign Key

Validity

Temperature

Time in Minutes

Weight in Gramme

Short Name

Callista

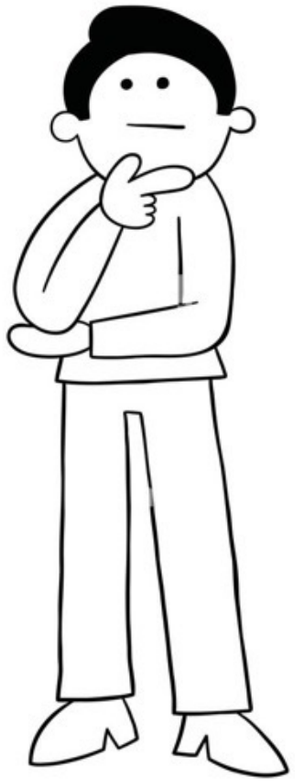
Live Demo


```
SQL> desc recipe

Name                 Null?       Type
___________________ ___________ ____________________________________
RECIPE_ID            NOT NULL    NUMBER DOMAIN PRIMARY_KEY
BEER_ID              NOT NULL    NUMBER DOMAIN FOREIGN_KEY
VALID_FROM           NOT NULL    DATE DOMAIN VALIDITY
VALID_TO             NOT NULL    DATE DOMAIN VALIDITY
START_TEMP                       NUMBER(3,1) DOMAIN MASHING_PROCESS
MASHING_TIME_1                   NUMBER(3) DOMAIN MASHING_PROCESS
MASHING_TEMP_1                   NUMBER(3,1) DOMAIN MASHING_PROCESS
MASHING_TIME_2                   NUMBER(3) DOMAIN MASHING_PROCESS
MASHING_TEMP_2                   NUMBER(3,1) DOMAIN MASHING_PROCESS
MASHING_TIME_3                   NUMBER(3) DOMAIN MASHING_PROCESS
MASHING_TEMP_3                   NUMBER(3,1) DOMAIN MASHING_PROCESS
MASHING_TIME_4                   NUMBER(3) DOMAIN MASHING_PROCESS
MASHING_TEMP_4                   NUMBER(3,1) DOMAIN MASHING_PROCESS
BOILING_TIME                     NUMBER(3) DOMAIN MASHING_PROCESS
HOP_TYPE_1           NOT NULL    VARCHAR2(20 CHAR) DOMAIN SHORT_NAME
HOP_QUANTITY_1                   NUMBER(3) DOMAIN WEIGHT_IN_GRAMME
HOP_TIME_1                       NUMBER(3) DOMAIN TIME_IN_MINUTES
HOP_TYPE_2           NOT NULL    VARCHAR2(20 CHAR) DOMAIN SHORT_NAME
HOP_QUANTITY_2                   NUMBER(3) DOMAIN WEIGHT_IN_GRAMME
HOP_TIME_2                       NUMBER(3) DOMAIN TIME_IN_MINUTES
YEAST                NOT NULL    VARCHAR2(20 CHAR) DOMAIN SHORT_NAME
MASHING_TYPE                     VARCHAR2(38) DOMAIN MASHING_PROCESS
```

Callista

~~Lessions Learned~~ Firsts Impressions



- Powerful new SQL feature – more than just “SQL Domains”
- Most of the functionality was possible before
- Typical use cases: Single Column Domains
- Few use cases for Multi Column Domains / Flexible Domains
- Single and Multi Column Domains cannot be combined
- Enumeration Domains are a bit tricky

Further Information

Oracle Documentation

- https://docs.oracle.com/en/database/oracle/oracle-database/23/sqlrf/create-domain.html
- https://docs.oracle.com/en/database/oracle/oracle-database/23/cncpt/application-data-usage.html
- https://docs.oracle.com/en/database/oracle/oracle-database/23/adfns/registering-application-data-usage-database.html

SQL Domains

- https://sql-99.readthedocs.io/en/latest/chapters/19.html
- https://www.postgresql.org/docs/current/sql-createdomain.html

Blog Posts

- https://www.salvis.com/blog/2024/12/24/evolution-of-a-sql-domain-for-semantic-versioning/
- https://blogs.oracle.com/coretec/post/less-coding-with-sql-domains-in-23c
- https://oracle-base.com/articles/23/domains-23


DRIVEN BY EXCELLENCE
Callista