```
--
--   ICS 212 Lab 05 - create schema
--


-- ================================================================================
-- Create cluster on orders, product description and product information.
-- ================================================================================
CREATE CLUSTER products (product_id NUMBER(6))
HASHKEYS 10000 SIZE 2048;


-- ================================================================================
-- Create customers table.
-- ================================================================================


CREATE TABLE customers
   ( customer_id       NUMBER(6)
   , cust_first_name    VARCHAR2(20)
   CONSTRAINT cust_fname_nn NOT NULL
   , cust_last_name     VARCHAR2(20)
   CONSTRAINT cust_lname_nn NOT NULL
   , nls_language       VARCHAR2(3)
   , nls_territory      VARCHAR2(30)
   , credit_limit       NUMBER(9,2)
   , cust_email         VARCHAR2(30)
   , account_mgr_id     NUMBER(6)
   , CONSTRAINT customers_pk PRIMARY KEY (customer_id)
         )
   PARTITION BY LIST(nls_territory)
         (
```

```
        PARTITION North_America VALUES('NA'),
        PARTITION Central_America VALUES('CA'),
        PARTITION Caribbean VALUES('CAR'),
        PARTITION West_Europe VALUES('WE'),
        PARTITION Africa VALUES('AF'),
        PARTITION Middle_East VALUES('ME'),
        PARTITION East_Europe VALUES('EE'),
        PARTITION South_Asia VALUES('SA'),
        PARTITION East_Asia VALUES('EA'),
        PARTITION South_East_Asia VALUES('SEA'),
        PARTITION Australia_South_Pacific VALUES('ASP')
        );

-- partitions on each region in the world
-- ================================================================================
-- Create warehouses table
-- ================================================================================

CREATE TABLE warehouses
   ( warehouse_id      NUMBER(3)
   , warehouse_name    VARCHAR2(35)
   , location_id       NUMBER(4)
   , CONSTRAINT warehouses_pk PRIMARY KEY (warehouse_id)
   ) ;

-- ================================================================================
-- Create table orders
-- Seperate orders by month with rotation for quick reference
-- ================================================================================
```

```
CREATE TABLE orders
   ( order_id          NUMBER(12)
   , order_date        DATE
   CONSTRAINT order_date_nn NOT NULL
   , order_mode        VARCHAR2(8)
   , customer_id       NUMBER(6)
   CONSTRAINT order_customer_id_nn NOT NULL
   , order_status      NUMBER(2)
   , order_total       NUMBER(8,2)
   , sales_rep_id      NUMBER(6)
   , promotion_id      NUMBER(6)
   , CONSTRAINT order_pk PRIMARY KEY (order_id)
   )
         PARTITION BY RANGE (order_date)
         (
         PARTITION pastdata VALUES LESS THAN (TO_DATE('01-JAN-2017','DD-MON-YYYY')),
         PARTITION Jan2017 VALUES LESS THAN (TO_DATE('01-FEB-2017','DD-MON-YYYY')),
         PARTITION Feb2017 VALUES LESS THAN (TO_DATE('01-MAR-2017','DD-MON-YYYY')),
         PARTITION Mar2017 VALUES LESS THAN (TO_DATE('01-APR-2017','DD-MON-YYYY')),
         PARTITION Apr2017 VALUES LESS THAN (TO_DATE('01-MAY-2017','DD-MON-YYYY')),
         PARTITION May2017 VALUES LESS THAN (TO_DATE('01-JUN-2017','DD-MON-YYYY')),
         PARTITION Jun2017 VALUES LESS THAN (TO_DATE('01-JUL-2017','DD-MON-YYYY')),
         PARTITION Jul2017 VALUES LESS THAN (TO_DATE('01-AUG-2017','DD-MON-YYYY')),
         PARTITION Aug2017 VALUES LESS THAN (TO_DATE('01-SEP-2017','DD-MON-YYYY')),
         PARTITION Sep2017 VALUES LESS THAN (TO_DATE('01-OCT-2017','DD-MON-YYYY')),
         PARTITION Oct2017 VALUES LESS THAN (TO_DATE('01-NOV-2017','DD-MON-YYYY')),
         PARTITION Nov2017 VALUES LESS THAN (TO_DATE('01-DEC-2017','DD-MON-YYYY')),
         PARTITION Dec2017 VALUES LESS THAN (TO_DATE('01-JAN-2018','DD-MON-YYYY'))
```

```
            );

-- ============================================================================
-- Create table order_items, which contains a concatenated primary key
-- ============================================================================

CREATE TABLE order_items
   ( order_id          NUMBER(12)
   , line_item_id      NUMBER(3)  NOT NULL
   , product_id        NUMBER(6)  NOT NULL
   , unit_price        NUMBER(8,2)
   , quantity          NUMBER(8)
   , CONSTRAINT order_items_pk PRIMARY KEY (order_id, line_item_id),
            CONSTRAINT order_items_order_id_fk FOREIGN KEY (order_id)
     REFERENCES orders(order_id)
   ) --;
            PARTITION BY REFERENCE (order_items_order_id_fk);
   --works if remove fk below and place constraint here
   --cannot place cluster and partition on a table
-- ============================================================================
-- Create inventories table, which contains a concatenated primary key.
-- ============================================================================

CREATE TABLE inventories
  ( product_id        NUMBER(6)
  , warehouse_id      NUMBER(3)
  CONSTRAINT inventory_warehouse_id_nn NOT NULL
  , quantity_on_hand   NUMBER(8)
  CONSTRAINT inventory_qoh_nn NOT NULL
```

```
  , CONSTRAINT inventory_pk PRIMARY KEY (product_id, warehouse_id)
  ) ;

-- ============================================================================
-- Create table product_information
-- ============================================================================

CREATE TABLE product_information
    ( product_id         NUMBER(6)
    , product_name       VARCHAR2(50)
    , product_description VARCHAR2(2000)
    , category_id        NUMBER(2)
    , weight_class       NUMBER(1)
    , warranty_period    NUMBER(5)
    , supplier_id        NUMBER(6)
    , product_status     VARCHAR2(20)
    , list_price         NUMBER(8,2)
    , min_price          NUMBER(8,2)
    , catalog_url        VARCHAR2(50)
    , CONSTRAINT product_information_pk PRIMARY KEY (product_id)
    )
          CLUSTER products (product_id);

-- ============================================================================
-- Create table product_descriptions, which contains NVARCHAR2 columns for
-- NLS-language information.
-- ============================================================================

CREATE TABLE product_descriptions
```

```
   ( product_id           NUMBER(6)
   , language_id          VARCHAR2(3)
   , translated_name      NVARCHAR2(50)
   CONSTRAINT translated_name_nn NOT NULL
   , translated_description NVARCHAR2(2000)
   CONSTRAINT translated_desc_nn NOT NULL
   , CONSTRAINT product_descriptions_pk
   PRIMARY KEY (product_id, language_id)
   )
         CLUSTER products(product_id);

-- ============================================================================
-- Create foreign key constraints now that all tables are in place.
-- ============================================================================

ALTER TABLE orders
ADD ( CONSTRAINT orders_customer_id_fk
    FOREIGN KEY (customer_id)
    REFERENCES customers(customer_id)
    ON DELETE SET NULL
   ) ;

ALTER TABLE inventories
ADD ( CONSTRAINT inventories_warehouses_fk
    FOREIGN KEY (warehouse_id)
    REFERENCES warehouses (warehouse_id)
   ) ;

ALTER TABLE inventories
```

```
ADD ( CONSTRAINT inventories_product_id_fk
     FOREIGN KEY (product_id)
     REFERENCES product_information (product_id)
   ) ;

--============================================================================
--this needed to be moved to the order items creation for the
--partition by reference can work correctly
--============================================================================
/*ALTER TABLE order_items
ADD ( CONSTRAINT order_items_order_id_fk
     FOREIGN KEY (order_id)
     REFERENCES orders(order_id)
     ON DELETE CASCADE
   );*/
--============================================================================

ALTER TABLE order_items
ADD ( CONSTRAINT order_items_product_id_fk
     FOREIGN KEY (product_id)
     REFERENCES product_information(product_id)
   ) ;

ALTER TABLE product_descriptions
ADD ( CONSTRAINT pd_product_id_fk
     FOREIGN KEY (product_id)
     REFERENCES product_information(product_id)
   ) ;
```

```
commit;


--customer email is easiest for contact and used mainly for logins
CREATE INDEX cust_email_idx
ON customers(cust_email);
--determine who is taking care of file for resolutions or auditing
CREATE INDEX account_mgr_id_idx
ON customers(account_mgr_id);
--shipping any product would require a name
CREATE INDEX cust_name_idx
ON customers(cust_last_name, cust_first_name);
--for statistical information based on sales
CREATE INDEX category_id_idx
ON product_information(category_id);
--to determine which product you are trying to reference / find
CREATE INDEX product_idx
ON product_information(product_name);
--used for inventory /sales purposes
CREATE INDEX product_id_idx
ON order_items(product_id);
--if a purchase come from a different country need to determine where
CREATE INDEX language_id_idx
ON product_descriptions(language_id);
--need to translate product information
CREATE INDEX translated_name_idx
ON product_description(translated_name);
--which of the many warehouses will be shipping the product
CREATE INDEX warehouse_name_idx
ON warehouses(warehouse_name);
```

```
--where the warehouse is located so you can ship from nearest one
CREATE INDEX location_id_idx
ON warehouses(location_id);
--when products were order for customer service inquiries
CREATE INDEX order_date_idx
ON orders(order_date);
--unique identifier to each customer could have over 1 million
CREATE INDEX customer_id_idx
ON orders(customer_id);
--to reference who sold the product
CREATE INDEX sales_rep_id_idx
ON orders(sales_rep_id);
--if there was a promotion on the product if so which one
CREATE INDEX promotion_id_idx
ON orders(promotion_id);
--id to determine warehouse in case there are multiple in close proximity
CREATE INDEX warehouse_id_idx
ON inventories(warehouse_id);
--loads of querys from customers /vendors
CREATE INDEX product_description_idx
ON product_information(product_description);

--============================================================================
--drop tables/cluster/indexes
--============================================================================
DROP TABLE order_items PURGE;
DROP TABLE orders PURGE;
DROP TABLE inventories PURGE;
DROP TABLE product_descriptions PURGE;
```

```
DROP TABLE customers PURGE;

DROP TABLE warehouses PURGE;

DROP TABLE product_information PURGE;

DROP CLUSTER products;
```